VOLUME III
SYSTEMS PHASE

CHAPTER 7
INTEGRATED AVIONICS SYSTEMS

SEPTEMBER 1985

USAF TEST PILOT SCHOOL
EDWARDS AFB, CA

TABLE OF CONTENTS

The information in Chapters 1, 2, 3, 5, and 6 were taken from MIL-STD-1553 Multiplex Applications Handbook, revised 1 June 1984, prepared by the Boeing Military Aircraft Development organization under U.S. Air Force Systems Command contract F33615-78-C-0112.

Chapter 1

THE MULTIPLEXED DATA BUS APPROACH TO AVIONICS INTEGRATION

1.1 Aircraft Avionics Integration

Avionics integration, which is defined here as the cooperative use of shared information among avionic subsystems, first became a necessity when avionics hardware requirements could no longer be met with independent and self-sufficient subsystems. Eliminating unnecessary duplication of information sensing and display, performance and reliability gains, cost reduction, and reduction of space requirements are usually given as the major reasons for integration. The avionics integration process typically began with the most complex subsystem because it had the most capability, as well as the most need for information from other subsystems. As digital technology progressed, this central subsystem was expanded to incorporate mission processing (processing not specifically associated with a subsystem or display).

Problems arose early in the centralization approach because even though the central computer concept made sensible use of shared information, subsystems were designed with no concern for interconnection with other subsystems. Each subsystem had been specialized, and the interfaces reflected this specialization. The central computer input-output (I/O) circuitry was designed to perform the functions of ordering this incoming and outgoing data, and the computer was often small compared with the size and complexity of the I/O.

It was reasoned that some of the centralization problems related to the complexity of the I/O could be solved if the I/O circuitry could by partitioned and distributed, alleviating the central unit's complexity. Commercial data transfer trends supported this view. Multiplexing makes information transfer convenient and simplifies I/O because the information transfer medium is reduced to a single wire pair. Sensors and processors are all "on the bus." This I/O philosophy was adopted extensively by military avionics integrators and made possible the distribution of the computation, permitting several computers to replace the more powerful central processor. These multiple computers added desirable redundancy features.

The integration approach using multiplexing is implemented by specifying the following:

a. information transfer formats
b. electrical interface characteristics.

Note that functions are accomplished by both hardware and software. Most of the problems associated with centralized I/O have been eliminated by this approach, and a decided improvement over previous approaches has been achieved. The modern integration approach uses multiple processors and buses to functionally partition the avionics along logical lines such as navigation, stores management, and communications. This functional partitioning eases the integration problem by allowing the subsystems to be developed relatively independently of each other prior to completing the total avionics integration.

1.2 Chronology of MIL-STD-1553

Development of a standard digital time-division multiplex data bus began in early 1968 and continued through 1978 with the latest revision (MIL-STD-1553B). The Society of Automotive Engineers (SAE), Aerospace Branch, established a subcommittee of industry and military personnel in 1968 to define some of the basic requirements of a serial data bus. By this means, an exchange of industry and military views was accomplished. The committee developed the first draft of a data bus standard that was similar to the present military standard. It represented a mixture of military standard requirements and procurement specification requirements. Its format allowed standardization on requirements that could be agreed upon, and a slash sheet in the appendix for requirements that appeared to be vehicle particular.

This document represented the best that the industry and the military could define at the time. The benefit of this document was that it produced a sounding board for ideas. In this respect, it was successful and provided the step forward required to develop the USAF military standard, MIL-STD-1553.

During the years from inception of the SAE committee to the release of the first military documents, the industry was designing and producing hardware for various multiplex systems. Some of these systems were developed prior to or during the standardization era (e.g., F-15 and B-1). Because of program timing, each system went its own way. However, with the production of the F-16, MIL-STD-1553(USAF) found its first full aircraft application.

By late 1974 and early 1975, the DOD directed the military to develop a single position and to make the necessary revisions to MIL-STD-1553(USAF). Based on this effort, MIL-STD-1553A was released in April 1975. Since 1975, industry and the military have continued to coordinate the standard through symposia, studies, and military development programs. With the standard available, the industry and the military began to apply the data bus to more operational vehicles and systems.

As applications became extensive, certain difficulties were recognized in MIL-STD-1553A. Discussions concerning these difficulties were conducted between SAE and DOD committees, resulting in the formation of an SAE task group in October 1976 with the assignment to develop suggested changes to 1553A. In October 1977, after review and discussion of suggested changes, recommendations were provided to DOD. MIL-STD-1553B was released as an official document on September 21, 1978.

1.3 PRACTICAL CONSIDERATIONS OF SYSTEMS INTEGRATION

Improvements in mission capability provided by integration can be classified into four categories:

a. Performance gain by the use of multiple subsystems to reduce the effect of error sources within a single subsystem; for example, Kalman digital filtering of Doppler and inertial navigation subsystem data can be used to obtain smoothing and prediction.

b. Reduction of total hardware by using a single sensor to provide common input data rather than dedicated input sensors for each subsystem.

c. Effective redundancy without massive hardware duplication made possible by the integration of identical, similar, or dissimilar sensors to make multiple sources of similar data available; this is called dissimilar sensor redundancy.

d. Reduced weight and increased flexibility of integration and test are achieved by using multiplexing.

The first three categories are common to any integration scheme, whereas the fourth category is associated with the multiplexing method of achieving integration. The serial, time-divison multiplexed data bus approach to avionics integration offers specific advantages over other methods.

Weight saving is achieved by the reduction of wire weight provided by the serial multiplexing of digital data as compared with the point-to-point interconnection required to achieve similar integration without the data bus. The data bus provides a path upon which many users can communicate with each other without requiring dedicated links.

The flexibility that is available in 1553 systems is one of its most important advantages. Because of the common serial interface, the high data rate (up to 50,000 words per second), the multiple access, and the command/response data format, the 1553 integration provides extensive flexibility in the development period as well as throughout the operational life cycle.

Another advantage of this approach to information transfer is the ability to control data flow in a scheduled manner from one location, namely the bus controller. Changes in the integration can be handled by message changes in the bus controller rather than by wiring and hardware changes to the subsystems. Also, the benfit of a synchronous schedule of data transfers can ensure data arrival when it is required and not on an asynchronous, uncontrolled basis.

Finally, the concept of multimission roles for a single airframe (or a restricted family of airframes) has become a major element in our military weapon planning. Changing threats make it necessary to plan for mission-adaptive and threat-adaptive avionics systems over the life of an airframe. In the past few years, it has been a goal of the DOD to develop and apply methods and technologies that would permit avionic systems to be developed which are easily constructed, modified, and operationally verified as mission needs change.

Two multimission concepts have emerged. One approach is to design a core set of avionics and peripheral avionics so the avionic suite can be readily changed by removing and replacing mission-dependent functions (peripheral avionics). Another approach is to depend on established interface standards (e.g., standard hardware and software modules) that permit an avionics system to be updated (retrofitted) throughout the life of the airframe. Both of these approaches represent ideal applications of multiplexing for integration.

This Page Intentionally Left Blank

Chapter 2

OVERVIEW OF MIL-STD-1553

The purpose of this chapter is to present the organization, scope, and contents of MIL-STD-1553 to assist readers who are unfamiliar it. The current version of 1553 (namely 1553B) will be discussed and key terms explained.

2.1 Scope of the Standard

A military standard is a document that establishes engineering and technical requirements for processes, procedures, practices, and methods. MIL-STD-1553, "Aircraft Internal Time Division Command/Response Multiplex Data Bus," has been widely used in integrated avionics designs since 1973. Each word in the title of the standard has a specific meaning:

a. Aircraft. This word denotes that the data buses are installed in DOD aircraft. Although there are instances of 1553 data buses used in ground applications, the characteristics of the bus have been determined by aircraft requirements.

b. Internal. This word denotes that the data bus is used for transmission only within an aircraft.

c. Time-Division Multiplex. The type of multiplexing for which the standard establishes requirements. Time-division multiplexing is the transmission of information form several signal sources through one communication system with different signal samples staggered in time to form a composite pulse train. A transmission line known as a twisted-shielded wire pair is used for transmission; consequently, all messages must be transmitted and received serially. The 1553 data bus is sometimes referred to as the 1553 serial data bus.

d. Command/Response. This denotes the communication protocol that distinguishes 1553 data bus operation from that of other data buses. Basically, the command/response protocol described in 1553 is a "speak only when spoken to" communication that is solely the responsibility of a designated controller.

2.2 Information Transfer Formats

The term "information transfer formats" is used in 1553B interchangeably with "message formats." The exchange of messages in 1553B is very precisely described, with 10 allowable formats (see fig. 2-1). If an exchange cannot be completed because of hardware or software failures, the standard specifies what is to be done. All methods of followup to retry the message or to determine the failure must be done within the allowable 10 message formats. It is this idea of proper exchange of messages that makes it appropriate to refer to them as "protocol" -- because it is similar to the process that diplomats use to exchange state notes. Message formats are composed of words, response time gaps, and intermessage time gaps.

Message formats are divided into two groups: mode commands and data transfers.

2.2.1 Mode Commands

Mode commands are those formats reserved for communication with the bus hardware and information flow management.

There is provision for 32 unique mode commands, and 1553B specifies the base 2 numbers that are to be used for 15 of these. The balance are reserved, which means the designer must secure special approval to use a reserved mode command number. However, the use of any or all defined mode commands is optional.

2.2.2 Data Transfers

Data transfer message formats, on the other hand, do not restrict the

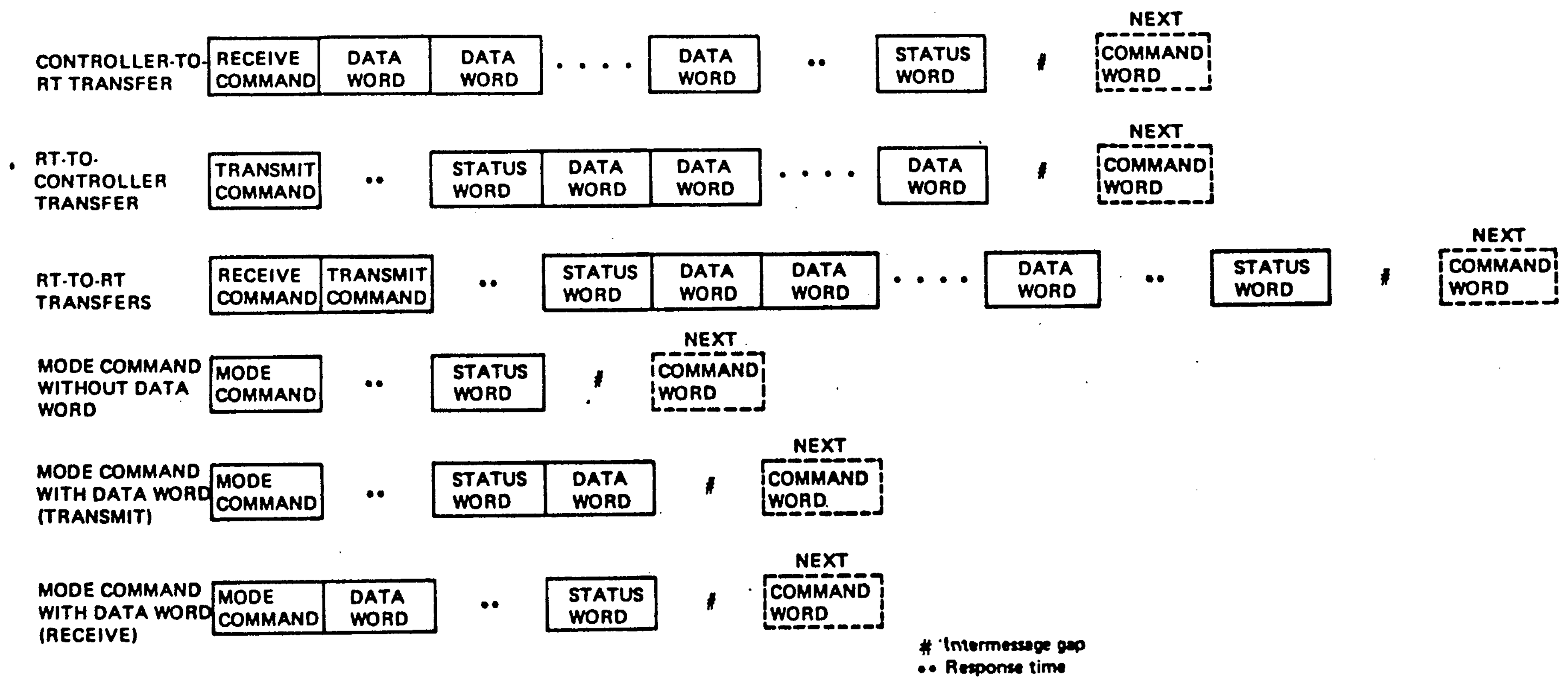


Figure 2-1. Information Transfer Formats

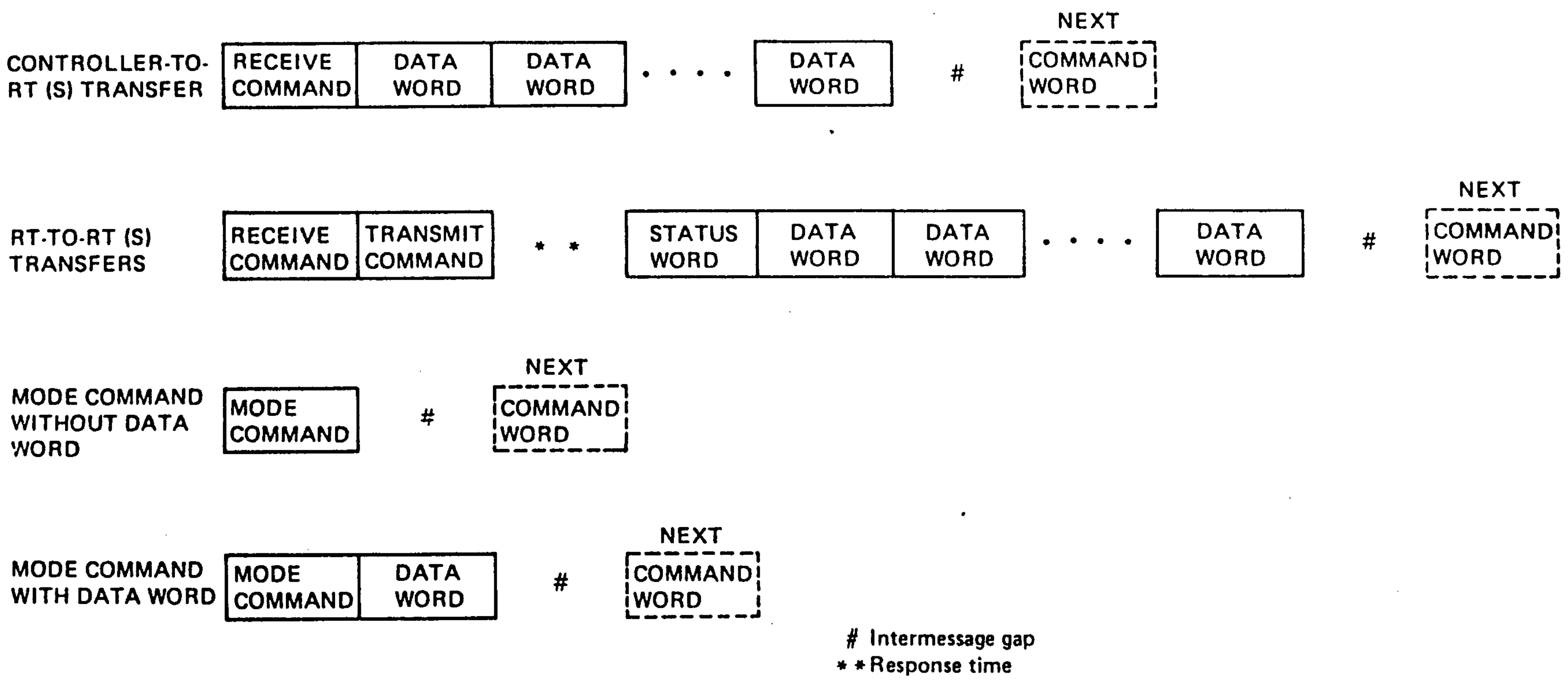


Figure 2-1 (cont'd). Information Transfer Formats designer to the same degree as mode commands. The restrictions are (1) no more than 32 words in any single message are to be used and (2) the most significant bits of any value or quantity will be transmitted first, with bits of descending significance following.

2.3 MIL-STD-1553 Terminals

A terminal is "the electronic module necessary to interface the data bus with the subsystem and the subsystem with the data bus..." There are only three functional modes of terminals: the bus controller, the bus monitor, and the remote terminal. The definition of a terminal as an electronic module should convey the notion of a piece of hardware that contains digital logic. Significant digital complexity is required because of 1553 response time and data storage specifications.

2.3.1 Bus Controller

The definitions section states that the bus controller is "the terminal assigned the task of initiating information transfers on the data bus." Other requirements are: (1) "The bus controller is the key part of the data bus system," and (2) "Sole control of information transmission on the bus shall reside with the bus controller... ." These quotes clearly define the bus controller.

2.3.2 Bus Monitor

The standard defines the bus monitor as "the terminal assigned the task of receiving bus traffic and extracting selected information to be used at a later time." Bus monitors are frequently used for instrumentation.

2.3.3 Remote Terminal

Any terminal that is not operating in either bus controller or bus monitor mode is operating in the remote terminal mode.

2.4 Types of 1553 Words

There are only three types of words: command words, status words, and data words. In 1553, a word is "a sequence of 16 bits plus sync and parity." The role of each of the three allowable word formats is as follows:

a. Command Word. This word is always used as the first word (or words) of a message. It will only be transmitted by a bus controller. This word defines the type of information transfer format that will be used.

b. Status Word. This word is always used as the first word that is transmitted by a remote terminal. (Bus monitors do not transmit at all.) This word contains the status of the transmitting remote terminal.

c. Data Word. This word (or words) is always transmitted contiquously with a command word, status word, and other data words.

Chapter 3

MULTIPLEX SYSTEM DESIGN

3.1 AVIONIC INTEGRATION DESIGN ACTIVITIES

System design begins with the statement of the requirements for avionic functions. The overall hardware and software requirements for a multiplex system will be derived from examination of functions and data requirements in the following areas:

a. Subsystems connected to the multiplex bus (or buses). What is connected to the bus? What are the data paths in the avionic system using the buses? What redundancy of data paths has been provided? What redundancy and/or isolation of function and equipment is required? Answers to these questions provide the overall context of avionics system operation.

b. Missions and modes of each mission. It is necessary to know the complete repertoire of missions, how these missions are supported by functions of the avionic systems, and what particular functions are to be performed during each phase of the mission. These groupings of functions by flight phase are called modes. (Note that these system or sensor modes are not related to MIL-STD-1553 "mode codes.") For example, the weapon delivery mode is usually distinguished from the waypoint navigation mode, even though navigation sensors may be used in each.

c. Functions of each sensor. Descriptions are needed for the inputs that each sensor requires (including sensor control information), what processing (computation) of sensor data is required for all avionic functions, and what data the sensor provides to other avionic systems. The sensor redundancy concepts and how data from redundant sensors will be used or reconciled need to be described, as well as sensor modes versus vehicle (avionic, weapon, flight control) modes. Thorough description of the interrelationship of sensors is required (e.g., inertial navigation update using another position-fixing sensor).

d. Functions of control and display. Descriptions are required for the overall interface of the controls and displays to the avionic systems, as well as which control and display functions depend on multiplexed data.

e. Other avionic functions. The advantages of multiplexing often are applied not only to the integration of sensors, processors, and controls and displays but also to more simple devices like switch positions, actuator positions, and power control. It is because of this application flexibility that the overall use of data in the system must be described.

The requirements derived from these considerations will establish the overall use of the 1553 data bus. It is quite likely that additional dedicated discretes will be used in an integrated system for critical functions (e.g., stores management enable or jettison). These interfaces need to be established and described at the same time the multiplex description is developed.

3.1.1 Functional Partitioning

The redundancy provided by the multiplex system is one of the key concerns and design requirements facing the system designer. The system designer must consider the following:

a. The basic level of redundancy within each subsystem.

b. The highest mission success probability associated with every function of each subsystem.

c. The isolation of each subsystem from other subsystems.

d. The independence of the redundant elements within each subsystem.

Based on this information about each subsystem being served and any added vehicle particular requirements (e.g., battle damage, no single failure can cause ... etc.), the system designer is able to establish a set of multiplex system requirements. These requirements will impact topology, multiplex control, and avionic control. The topology is usually the most visible point in the multiplex system where redundancy can be observed. However, the system designer must consider much more than just having a topology that meets the observable redundancy requirements. The redundancy of the bus controller and its associated circuitry involved in the detection and correction of a failure as well as the same functions in a remote terminal are all part of meeting the redundancy requirements of the integration.

3.1.2 Data Bus Topology

Data bus topology is the map of physical connections of the data bus terminals to the data bus. It includes all terminals and data buses involved in the data bus integration of the vehicle. Data bus topologies can be categorized into the following two general categories:

a. Single level
b. Multiple level

A single level bus topology is the simplest bus topology and is exemplified by the F-16 avionic bus architecture (see fig. 3-1). In a single-level bus topology, all terminals are interconnected via the same data bus. The redundancy requirements of a particular application may require a single-level topology to be implemented using multiple interconnecting cables operating in various modes (active or passive). However, the requirements to use multiple buses for redundancy purposes does not change the single-level bus topology definition if the following criteria are maintained:

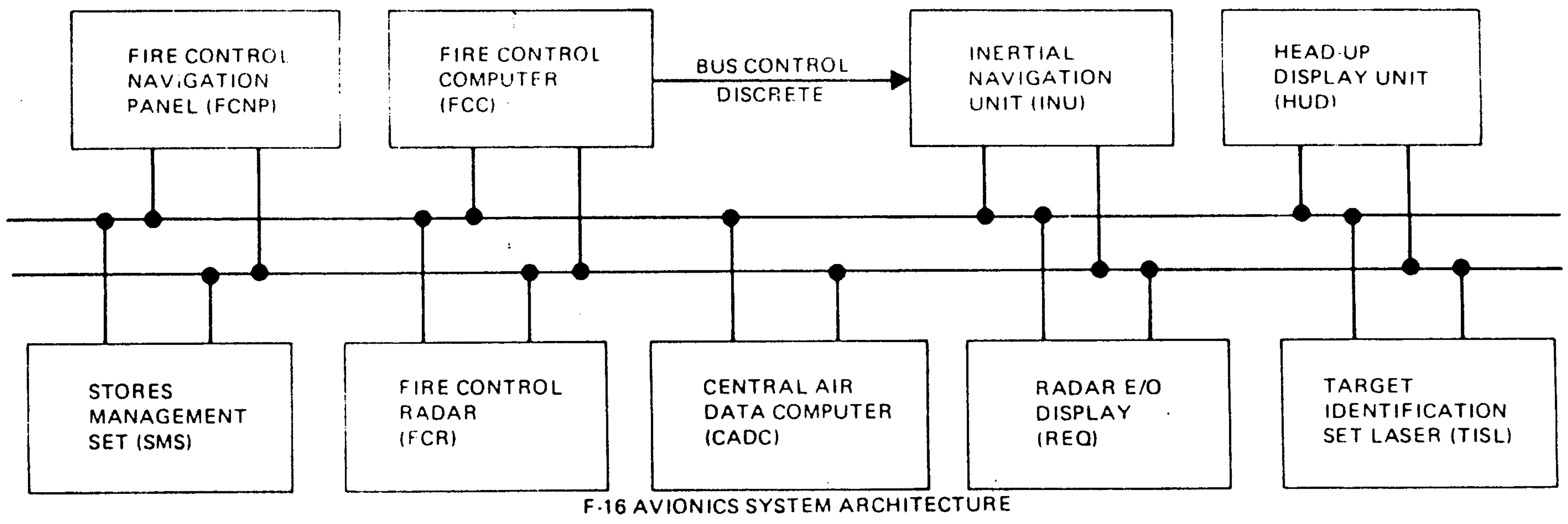

FIRE CONTROL NAVIGATION PANEL (FCNP)
FIRE CONTROL COMPUTER (FCC)
BUS CONTROL DISCRETE
INERTIAL NAVIGATION UNIT (INU)
HEAD-UP DISPLAY UNIT (HUD)
STORES MANAGEMENT SET (SMS)
FIRE CONTROL RADAR (FCR)
CENTRAL AIR DATA COMPUTER (CADC)
RADAR E/O DISPLAY (REO)
TARGET IDENTIFICATION SET LASER (TISL)
F-16 AVIONICS SYSTEM ARCHITECTURE

Figure 3-1. F-16 Avionic System Architecture

a. All terminals are connected to each data bus cable

b. Communication on each data bus is identical

The methods of bus control and the redundancy communication techniques used are peculiar to the application and will be discussed under these areas.

The multiple-level bus topology is an expansion of the single-level topology and can be expressed in two basic forms:

a. Multiple levels of buses with equivalent levels of control

b. Multiple levels of buses with hierarchical levels of control

The multiple-level bus topology with equivalent levels of control is exemplified by weapon systems that use multiple, single-level bus topologies for different functions. An example of this relationship are the B-1 EMUX, AMUX, and CITS (see fig. 3-2), that represents three single-level bus topologies with interconnections for data exchange purposes only, thus producing a multiple-level bus topology for the vehicle. Each of these single-level bus topologies operates independently of the others with equivalent levels of control. Another method of achieving a multiple-level bus topology within a subsystem integration is exemplified by the B-52 OAS (see fig. 3-3) multiple-level bus topology, which is partitioned into two single-level topologies (i.e., control and display versus navigation and weapon delivery). This trend to multiple, single-level topologies within a vehicle or in a large subsystem integration is a natural evolution of the single-level bus topology.

A second form of multiple-level bus topologies occurs when one or more single-level bus topologies are integrated with another single-level bus topology where the levels have a control relationship (see fig. 3-4). The bus level inequality may be expressed as follows:

a. Local buses, subordinate (under submission to global bus)

b. Global bus, superior (control over local buses)

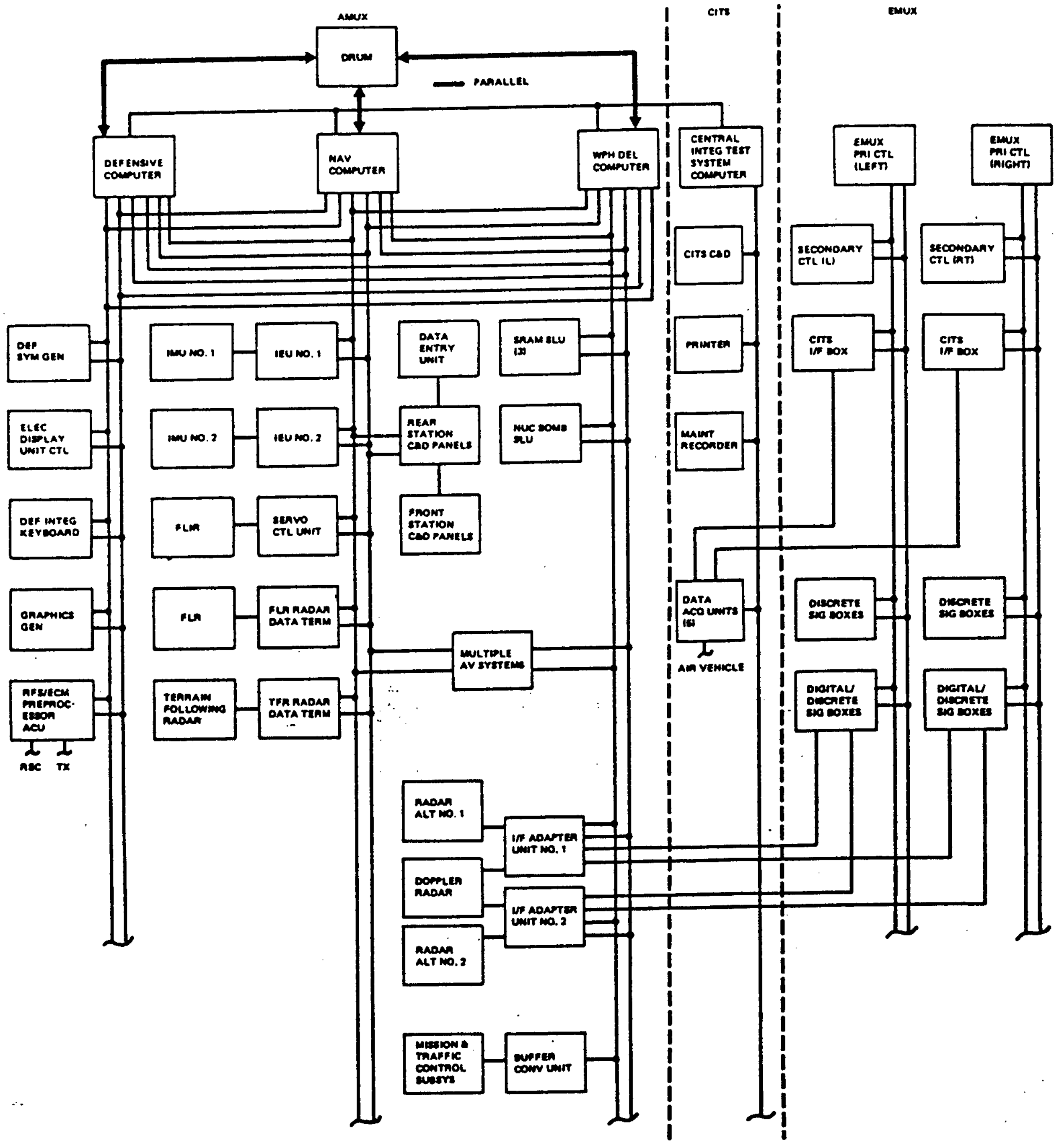


Figure 3-2. B-1 Multiplex System Architecture

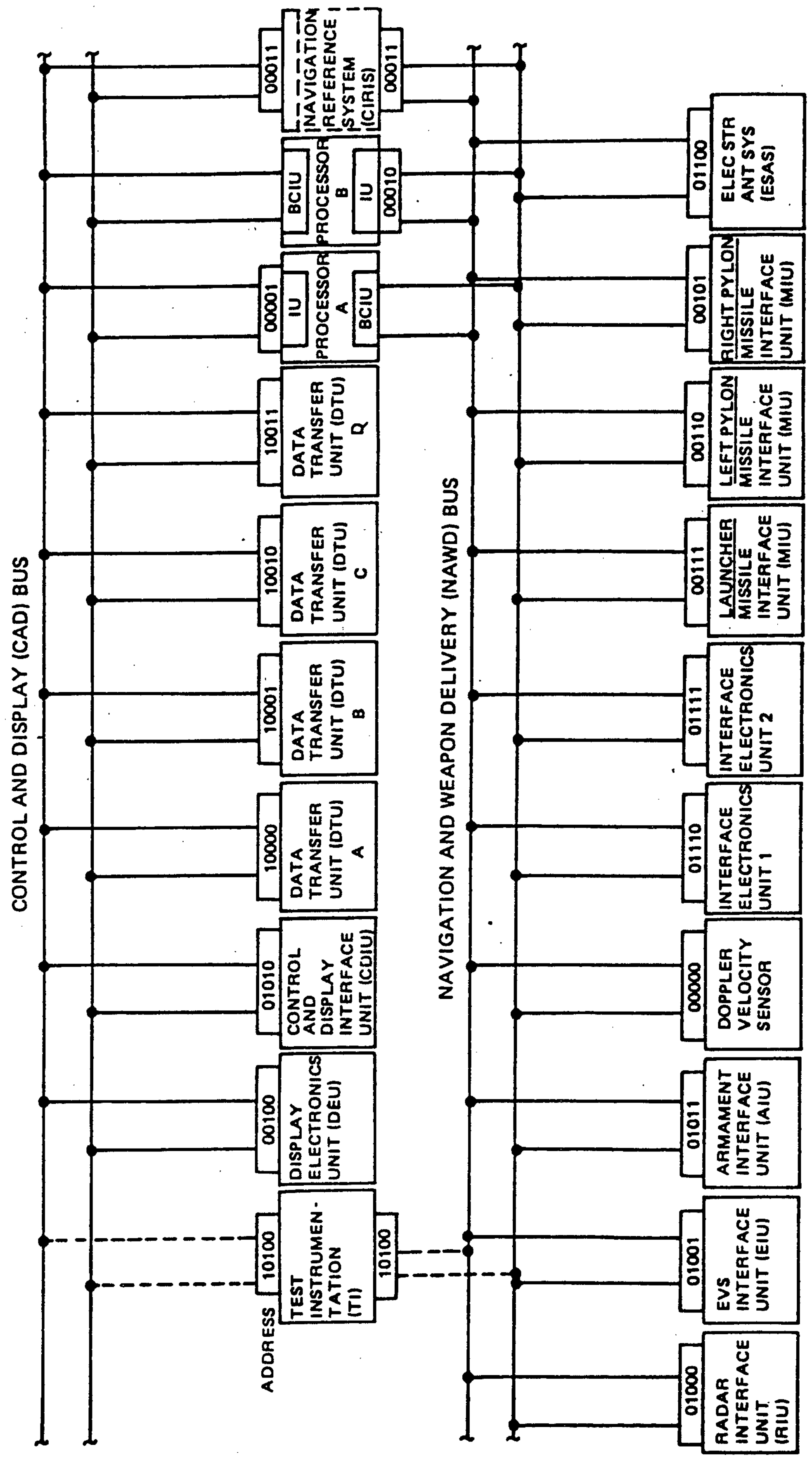
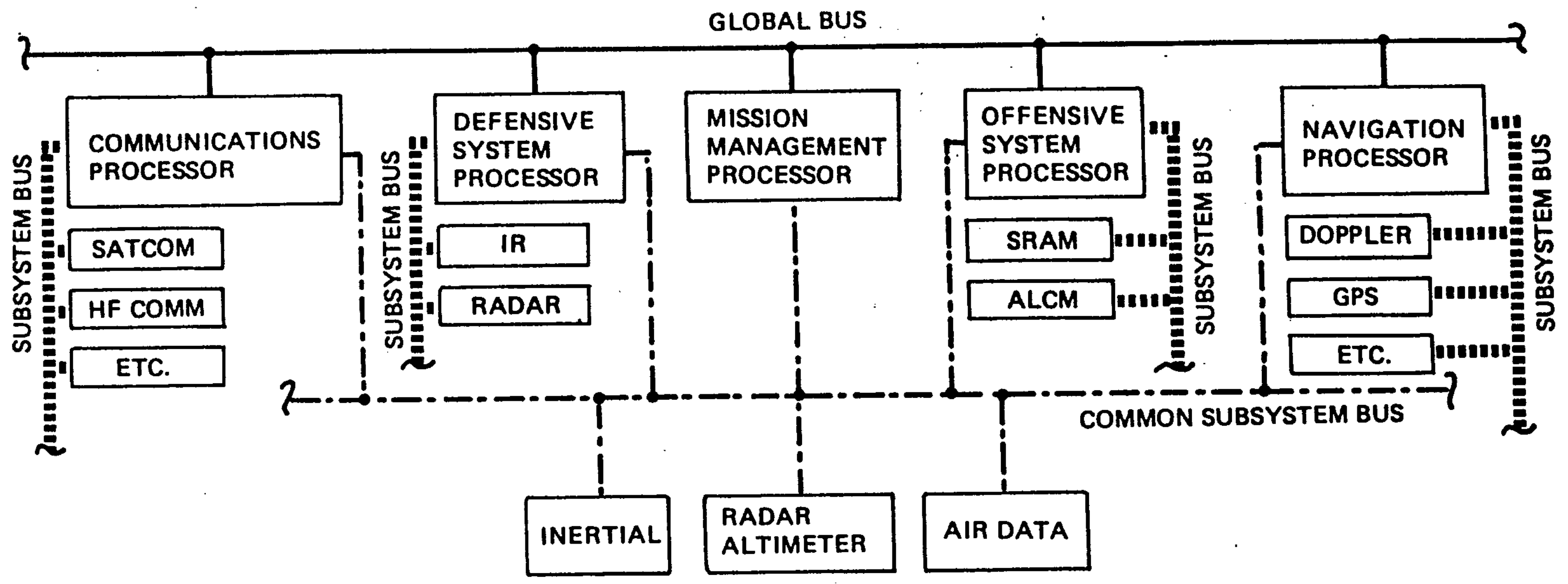

CONTROL AND DISPLAY (CAD) BUS
NAVIGATION AND WEAPON DELIVERY (NAWD) BUS
ADDRESS
10100 TEST INSTRUMENTATION (TI) 10100
00100 DISPLAY ELECTRONICS UNIT (DEU)
01010 CONTROL AND DISPLAY INTERFACE UNIT (CDIU)
10000 DATA TRANSFER UNIT (DTU) A
10001 DATA TRANSFER UNIT (DTU) B
10010 DATA TRANSFER UNIT (DTU) C
10011 DATA TRANSFER UNIT (DTU) D
00001 IU PROCESSOR A BCIU
BCIU PROCESSOR B IU 00010
00011 NAVIGATION REFERENCE SYSTEM (CIRIS) 00011
01000 RADAR INTERFACE UNIT (RIU)
01001 EVS INTERFACE UNIT (EIU)
01011 ARMAMENT INTERFACE UNIT (AIU)
00000 DOPPLER VELOCITY SENSOR
01110 INTERFACE ELECTRONICS UNIT 1
01111 INTERFACE ELECTRONICS UNIT 2
00111 LAUNCHER MISSILE INTERFACE UNIT (MIU)
00110 LEFT PYLON MISSILE INTERFACE UNIT (MIU)
00101 RIGHT PYLON MISSILE INTERFACE UNIT (MIU)
01100 ELEC STR ANT SYS (ESAS)

Figure 3-3. OAS Multiplex System Architecture



- Isolated, independent, integrated subsystems
- Independent redundancy and control
- Fault-tolerant reconfiguration
- Standardized bus structure

Figure 3-4. Hierarchical Multiplex Architecture

The primary reason for the difference in control is based on functional usage and not on the interconnection of the terminals. Data bus topologies can also be defined on the basis of interconnection requirements of the user terminals. These requirements would also establish the interconnection of a sensor to a local or a global bus.

3.2 Data Bus Control

The control philosophy used to maintain the communication on a data bus is described MIL-STD-1553. The control approaches are of two types:

a. Stationary master
b. Nonstationary master

The stationary master bus control concept is used when a single bus controller orchestrates the bus communication for all devices on that data path. Only in the event of a failure of the bus controller hardware or software will another bus controller (backup bus controller) operate the data bus. Obviously, as discussed in the topology section, multiple stationary master bus controllers can exist within a system, each controlling its own data bus.

The nonstationary master bus control concept is used when more than one bus controller orchestrates the bus communicaton for devices on that data path. MIL-STD-1553B provides a method of transferring control from an active bus controller to a potential bus controller (dynamic bus control mode code). This mode code provides a protocol format for issuing the bus controller offer with the responding status word providing an acceptance or rejection of the offer. Since the military standard prevents the operation of multiple bus controllers simultaneously, a method must be established to determine when the above mode code is to be issued and "to whom" it should be offered. The development of the timing (when) and the ordering (how the selection is achieved) is not specified by the standard and must be established by the system design.

3.3 SOFTWARE DESIGN

This section deals with the techniques and considerations of software design for an avionic system using 1553 data buses. The discussion is concerned with the software that controls messages on the bus rather than application software. The term "control software" includes the avionic system executive, bus control, and error handling. The term "application software" includes such functions as navigation (dead reckoning, aided inertial dead reckoning, navigation sensor management) fire control, weapon delivery, and communication control. The interface of application functions with system control is included and is discussed from the point of view of segregating all supervisory functions from application software. Obviously, the multiplex system software must support the performance of all required avionic system functions.

3.3.1 SYSTEM CONTROL CONSIDERATIONS

The software designer of a multiplex system is typically faced with the task of determining the total software requirements concurrently with system and hardware design. The software engineer must obtain the requirements for software (of which the system control will be least obvious) from several sources. "System control" means both multiplex system and avionic system monitoring for correct operation, error handling, and failure handling.

The 1553 standard requires the bus controller to initiate all data bus transfers. The question of "who is in control" is very important. Since the bus controller operation is the single point of avionic and multiplex system control, it should meet the redundancy requirements of the entire system. Answers to the following questions relating to the approach for bus control are essential: (1) What is the overall approach to controlling the data transfers of the avionics system? (2) What is the concept of redundancy with respect to the bus control? Is it a duplicate capability, an alternative control capability, or an alternative controller managing limited, degraded, or backup capability?

The requirements of avionic system control cannot be defined without clear definition of the weapon system functions, the sensors, the avionic architectures, and the functions allocated to or expected of software. The following sections discuss control considerations from three interrelated points of view: data transfer control, multiplex system control, and errors and hardware failures.

3.3.1.1 Data Transfer Description

It is necessary to define all avionic system data transfers that are implemented using the multiplex bus. An examination of these will establish control requirements. Each individual data transfer must be defined in terms of the following attributes for each mode:

a. Source function
b. Destination function
c. Data definition in 16-bit words
d. Iteration rate if it is periodic data
e. Conditional events if it is aperiodic data
f. Allowable latency
g. Conditional events related to the data paths
h. Error response characteristics

Determining the source and destination of each data item is the first step in the definition of the input and output of the functioning avionic system. Included in this description is the definition of the data in terms of 16-bit words because that is the medium of transmission over the bus. A data item may be an input to different functions according to the particular phase of the mission and may be transmitted at different frequencies.

Most multiplexed avionic systems operate on fixed schedules of data transfers. The requirements for the scheduling come from the examination of the largest and smallest minimum iterations and allowable latencies. The slowest iteration rate, which is the least common multiple of the faster iteration rates, is normally defined as the major cycle (see fig. 3-5). Over the course of a major cycle, all periodic transmissions and all

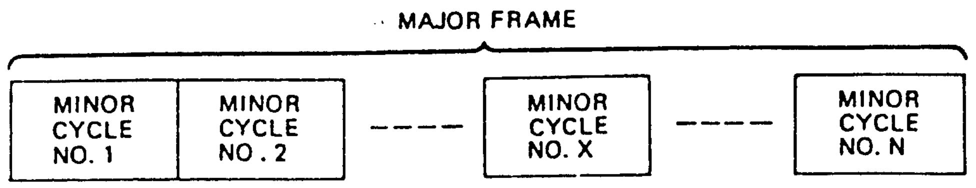


Figure 3-1. Major Cycle

periodic computations occur at least once. The minor cycle is normally the frequency of the most rapidly transmitted periodic data. Typical major frames are 1 second in length, while minor frame lengths can be binary (2^N/sec) or decimal (10^N/sec) with common values being 1/128, 1/64, 1/50 sec, etc.

For example, if the major frame is 1 sec long and there are 64 (2^6) minor cycles, then each minor cycle is 1/64 seconds or 15.625 ms long. Each periodic message would occur at least once each major frame, up to a maximum of 64 times. If a transaction needed to occur eight times per second, it must occur during one of the first eight minor cycles (64/8 = 8) and every eight minor cycles thereafter.

In the example of a transaction occurring eight times per second, shown in Table 3-1, if the first transaction occurred in minor cycle 3, later transactions would occur in minor cycles 11 (i.e., 8 + 3), 19, 27, 35, 43, 55, and 63.

Aperiodic messages, while rare, are based upon conditional events, such as a requirement to present a display to a crewmember within X-milliseconds of keyed-in commands, or a requirement to acquire data in a data buffer before it is lost to the next input to the buffer. The latter case is typical for keystrokes from keyboards.

3.3.1.2 Multiplex System Control

MIL-STD-1553 requires that the multiplexed data transfers be initiated by a bus controller and be followed by a response from a terminal in normal operation. The only exception is that a bus controller may command a broadcast, which does not require a response. In all cases, the actual transmission is either a combination of protocol data and avionic data formatted into 16-bit words or it is a transmission of protocol information without avionic data.

The types of data transfers and the implications for the software designer are as follows:

a. Remote terminal to bus controller. This type of data transfer is used to provide data to the bus controller. The bus controller is almost always a mission computer, fire control computer, navigation computer, etc. It requires data from several sources, such as an air data sensor, inertial measurement unit, radio navigation sensor, etc., to perform its assigned computational functions. Therefore, it usually will also be assigned the function of bus controller and will initiate the requests to the remote terminals for the data that it needs. The data needs of the mission computer establish the requirements for data transfers to it from other sources on the bus.

b. Bus controller to remote terminal. Typically these types of data transfers are related to the role of the processor that has the bus control function. A mission computer may have the requirement to be the data source to devices, providing such data as position update to an INS, or the requirement to transmit display parameters to a graphics generator. The mission computer often serves as the processor to effect weapon system control, such as fire control, in which case it is controlling both the multiplex system and computing parameters for target designation, weapon initialization, etc. In this case, controller to remote terminal transfers are data transfers from the fire control computer to the remote terminals that contain the interfaces to target designators, stores, etc. These types of data transfers may also be used as a method of central distribution in which data are taken from a remote terminal, reformatted and retransmitted to other locations.

c. Remote terminal to remote terminal. The bus controller does not need to receive and retransmit all data even though it is in control of the bus. An important class of data transfers is the direct transfer of data from one remote terminal to another, which can be used if the processor that contains the bus controller is not involved in the processing of the data and if reformatting is not required. In avionic systems that employ more distributed processing (e.g., CADC, INS on the bus) the additional processing capability at those remote terminals can be used to select and format data for direct remote-terminal-to-remote-terminal data transfers.

d. Broadcast. The broadcast data transfer is an option of 1553B but not currently in general use in military avionics. Broadcast allows the simultaneous transmission of the same data to more than one remote terminal. This transfer format may be used for avionic data transfers when significant reduction in processing or bus message traffic is needed and the command/response validation feature of each message is not required. For example, broadcast of roll and pitch data for aircraft flight-path control to a dual-, triple-, or quad-redundant flight control system may serve to simplify both avionics and flight control software. Note that broadcast does not eliminate the need to determine status (e.g., message received) but that status must be determined by separate requests of the controller to remote terminals.

Each unique data transfer must ultimately be identified to the multiplex system hardware and software by its unique combination of terminal address and subaddress. It is this feature that establishes the requirement that the data transfers be organized into messages. It is common in the system design process to prepare tables that define, for each subsystem on the bus, the complete data transfer specification into messages. Entries in the table are usually as follows:

a. Subsystem name, for example, INU, CADC, radar.

b. Subsystem terminal address.

c. Data block ID.

d. Subaddress.

e. Word count.

f. Refresh rate, for example, the rate at which the subsystem updates a variable.

g. Transmit rate, usually stated as a minimum value, at which the subsystem will be requested to transmit the data.

Separate tables are required for transmit and for receive for each terminal, whether it is a remote terminal or a bus controller. The software designer must define all data blocks that will be transmitted or received under all system conditions, normal or abnormal. The control software will handle the data according to the data block definition. Therefore, an exact correspondence between the input and output of data blocks and the use of the data must exist.

Multiplex system control and avionic system control are accomplished in software via the executive. The executive manages the multiplex elements in both normal and error conditions. Normal operating conditions include:

a. Initialization of the bus.

b. Effecting the transmission of messages.

c. Setting up of proper output message sequences.

d. Timing for periodic message sequences.

e. Aperiodic message setup, transmission, and/or reception.

f. Communication of time event or message event (arrival) to an application task controller or to the recipient task.

g. Transfer of control.

h. Reconfiguration due to change of mission mode.

Abnormal operating conditions include:

a. Handling transmissions errors.

b. Responses to failure of subsystems on the bus.

c. Failure record-keeping.

d. Reconfiguration because of failure.

3.3.1.3 Errors and Hardware Failures

Provision must be made during system design to handle errors in data transmission, power transients, hardware failures, and data errors. Determining the requirement for a response to a detected error is difficult, because there is no guidance in 1553 and no well-accepted guidelines for doing an analysis that shows that one set of responses is superior to another if mission success probability is the measure. If mission success probability is computed for several candidate responses to detected errors, only those actions that increase the probability should be considered for implementation. Laboratory investigation (such as in a hot bench) may be highly desirable to determine both the effect of a response and the cost in software to get the response.

The 1553 data bus does provide superior error detection capability for messages intended to be transmitted and received. This does not mean that inherent errors in data are also detected. Therefore, a software engineer should include data reasonableness checks or other authentication before data are used. This is particularly important for any data that are critical to mission success.

This Page Intentionally Left Blank

Chapter 4

EVOLUTION TOWARD A MULTI-BUS ARCHITECTURE FOR ARMY HELICOPTER AVIONIC SYSTEMS

Over the past decade the architecture of Army helicopter avionic systems has evolved from stand-alone subsystems to integrated subsystems characterized by digital data buses and embedded microprocessors. As these first generation integrated systems are reaching maturity, the U.S. Army Avionics Laboratory is turning its attention to some of the issues which must be addressed in preparation for the development of future systems. This paper presents an evolving concept for a data bus and processing structure which is being developed under an in-house Avionics Laboratory technology base effort.

4.1 Avionics Architecture Requirements

As a result of the need for significantly increased integration of aircrew functions to meet the demands being placed on Army Aviation, it has become apparent that a technical requirement exists for an architectural concept with a number of characteristics that go beyond that which is in development today. For example, many of the subsystems that come together to synthesize a total helicopter avionic system are developed by organizations with specific expertise in various functional areas (e.g., navigation, flight control, weapons, target acquisition, etc.). These organizations are usually both geographically and managerially separated, both in the government (customer) and in private industry (supplier). It has become apparent that an architectural concept is required that will

by Dr. Joseph A. Dasaro, U.S. Army Avionics Research and Development Activity, Fort Monmouth, New Jersey.

allow these various functional areas to develop both subsystem hardware and software independently and still conform to an architecture that facilitates future total system integration. Similarly, subsystem requirements in the control/display area must be developed with the subsystem. However, the architectural concept must allow subsystem control/display functions to be mapped into the total helicopter control/display system without re-doing the associated software. From the subsystem developer viewpoint, the system architecture must be such that at the time of total system integration, subsystem specific hardware (e.g., sensors) must be easily incorporated into the bus structure and the system level software absorbable into the system processing elements.

Another desirable feature or characteristic of the new architecture would be a means for high speed interface between processing elements of the system. Additionally, both the processing structure and bus structure must lend itself to fault tolerant designs.

Finally, there are two remaining areas which the architectural concept must address if it is to be widely accepted. First, the hardware implementation of the concept must be such that it lends itself to a procurement process that does not unnecessarily restrict the government and contractors to either each other or specific technology; and second, the processing elements must be such that standardization can be achieved on a foundation of structured programming and a higher order language.

Considering all of the above characteristics, an architectural concept is evolving based on multi-buses and multi-processors.

4.2 Evolution To Date

It is most helpful prior to presenting a proposed architecture to first look back and review the thought processes that led to our current architecture.

An accepted beginning to this era of integrated system architectures is the promulgation and acceptance of the U.S. MIL-STD-1553 data bus (NATO STANAG 3838). For ease of discussion, let the line depicted in Figure 1 represent such a dual (redundant) twisted shielded pair data bus. The Army Avionics Laboratory first used the bus concept to integrate the aircraft communication, navigation, and identification (CNI) equipment (see referenced paper). Architecturally speaking, the integration of this equipment can be represented by Figure 2. In general, all processing was accomplished in 8-bit microprocessors embedded in Remote Terminals (RT's) which also served as interface media to the mostly existing inventory radio equipments. One or more control/display units (CDU's) provided means for crew interface. Architecturally, the system can be characterized as of low bus data rate, modest level of fault tolerance (e.g., completely redundant processing), and little synergism. The software was in assembly code and generally not portable. Spinoffs of this system developed by the Army are currently being used in a large variety of aircraft (such as C-130, SRR, MRS, A-10, KC-135, F-111, EA6B, etc.).

The next step in system evolution was an expansion to include helicopter functions such as flight displays, engine displays, caution/warning/advisory subsystems, electrical systems (circuit breakers), and a large number of controls/displays referred to as secondary systems. This step was accomplished by addition of the items shown in Figure 3. The crew interfaces for the functions absorbed are four multi-function displays (MFD's) and two keyboard terminal units (KTU's). Eight remote terminals provide interface to the many hundreds of aircraft sensors, transducers, etc. Generation of alphanumerics and vector graphics are accomplished in two programmable symbol generators (PSG's). The processing elements consist of two Sperry SDP-175 16-bit processors. Programming is still generally in assembly code; however, concepts such as structured software with software modules are evolving. Architecturally, the system can be described as of modest data rates, fully redundant, and through use of dynamic bus allocation (DBA) somewhat partitioned software (i.e., for a fixed time period of every frame the master bus controller in one of the SDP-175's

Figure 1. "The Beginning" - STANAG 3838 Data Bus (MIL-STD-1553)

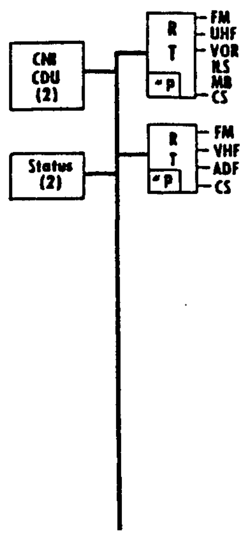


Figure 2. CNI Subset: The First Step

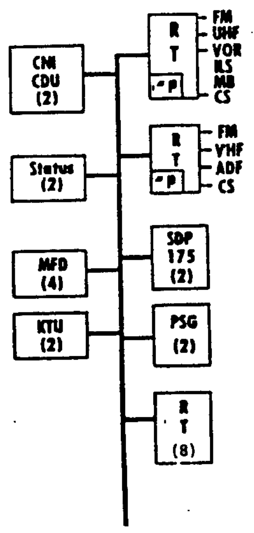


Figure 3. Expansion to Include Flight/Engine/Caution/Warning, etc.

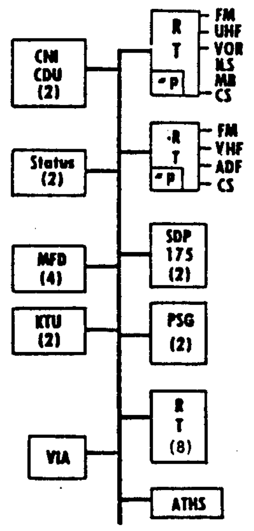


Figure 4. Voice Interactive Avionics and Airborne Target Handoff System Added

relinquishes the bus to the CNI subset at which time all CNI traffic is accomplished). This architecture (with some hardware optimization which eliminates the need for DBA) is currently being applied to systems such as the OH-58D and the HH-60D. The Army Avionics Laboratory (see referenced paper) is now adding Voice Interactive Avionics (VIA), both synthesis and recognition, and a digital data link, the Airborne Target Handoff System (ATHS), to the system (see Figure 4).

4.3 The Near Future

In addition to adding these new functions, it has become apparent that through additional processing a much higher level of automation can be introduced into this system. If it is assumed that this new processing is to be implemented using the new DOD standard Higher Order Language (HOL), Ada, and the processor used is to be characterized by a standard Instruction Set Architecture (ISA), such as MIL-STD-1750A for 16-bit machines or MIL-STD-1862 for 32-bit machines, then the addition of a processor as shown in Figure 5 becomes very appealing. (For illustrative purposes only one processor is shown; however, in reality there would be redundancy.) Now further assuming that all the necessary software tools are mature enough to distribute to the engineering elements that have cognizance over the subsets of this system, the processing functions currently embedded in the CNI subset and the SDP-175's can be placed in software modules which can then reside in the new processor. Only a modest amount of processing would remain outside this new processor (input/output, signal conditioning, symbol generation, etc.).

If attention is now turned to the architecture evolving in the navigation technology area, a virtual step-by-step analogy will occur resulting in expansion of the system to that shown in Figure 6. Similar to the basic aircraft area, by the addition of processing, a much higher level of synergism can be achieved among the navigation elements. The processor shown at the bottom of the navigation subset data bus in Figure 7 is, of course, identical to the basic aircraft processor just discussed (HOL, ISA,

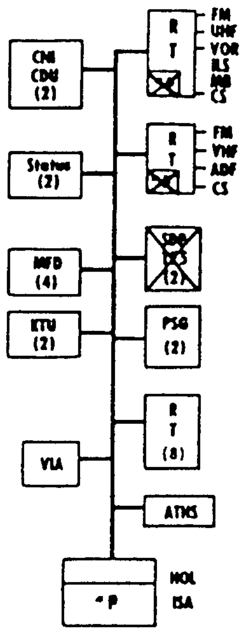


Figure 5. A Processor Characterized by Standard HOL and Standard ISA

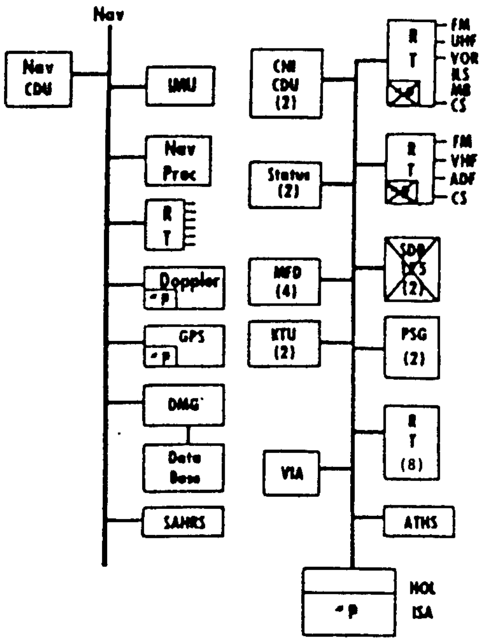


Figure 6. Navigation Technology Architecture

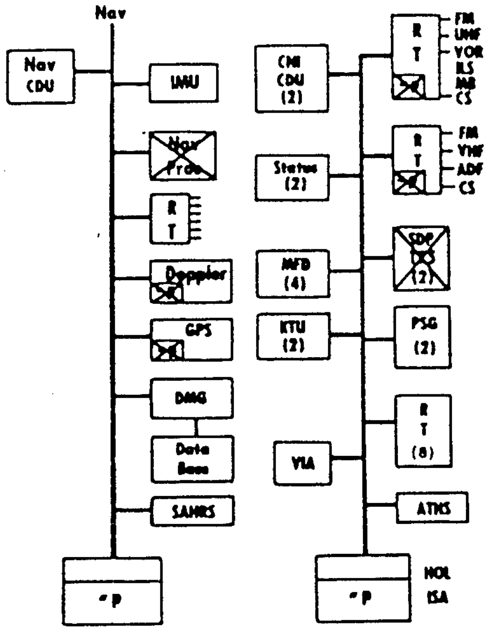


Figure 7. Navigation System Processing

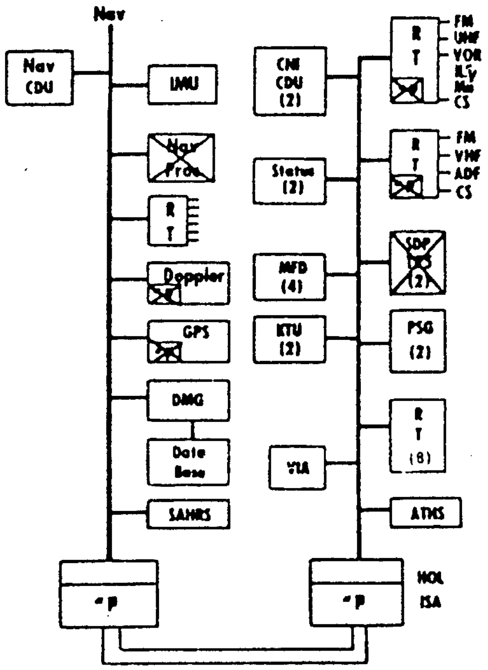


Figure 8. High Speed Bus

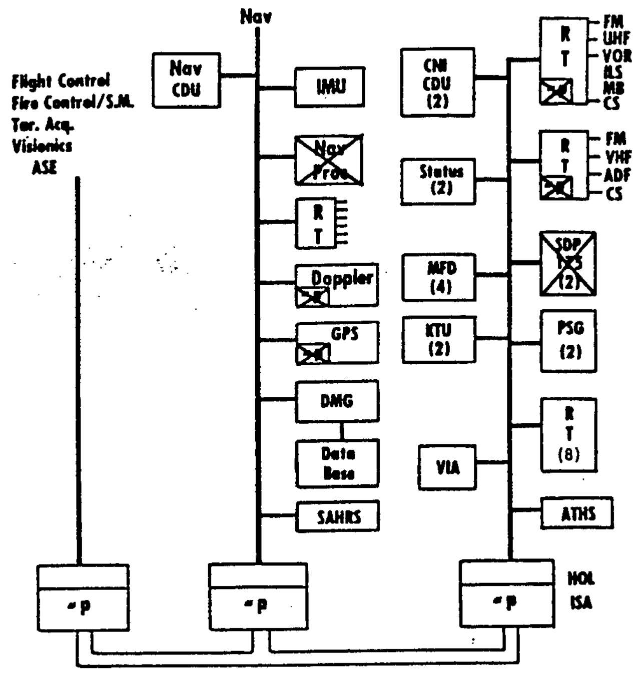


Figure 9. A Further Analogy

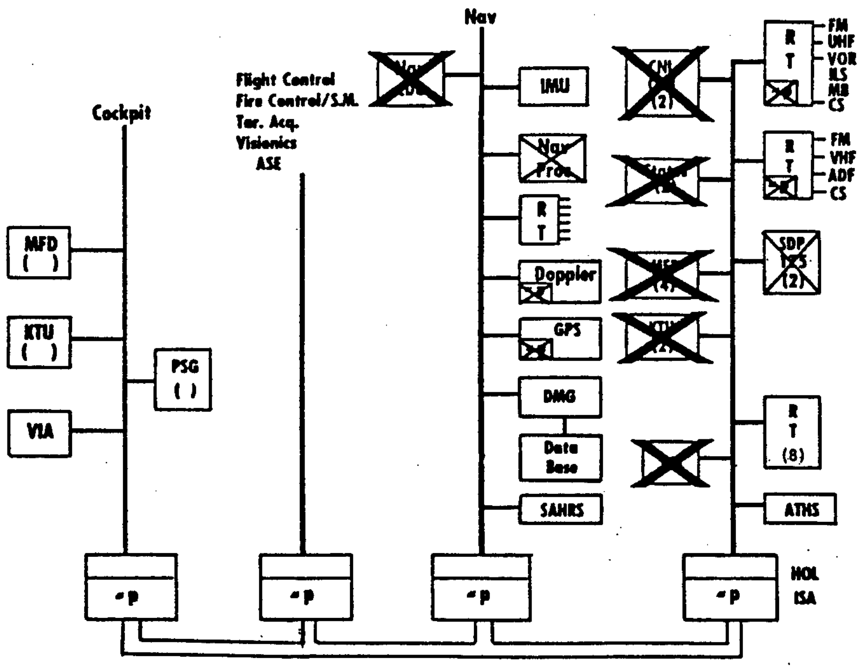


Figure 10. Mapping Crew Interface Functions from Subsystems to Total System

structured programming, etc.). In actual fact, they could, of course, be the same processor or maybe two processors residing in the same box interconnected by a high speed bus (see Figure 8). Nevertheless, the important point at this time is to note that all system level processing software will be in a higher order language (Ada) and will be in modular form.

A further analogy (Figure 9) can now be made for the system level processing in the various functional areas such as flight control, fir control, stores management, target acquisition, visionics, aircraft survivability equipment, etc. That is to say, in each of these areas, system level processing functions would be developed using the same software tools and the same software structure.

Now each of these functional areas will require unique crew interface actions which, although developed separately, must be mapped into the total cockpit. During development of these various subsystems, some crew interface software and hardware must be developed and used to accomplish many of the necessary steps in subsystem development. It is important that in this subsystem development process the control/display hardware used and the software developed for it be capable of being mapped into a totally integrated cockpit (i.e., minimize hardware uniqueness/use software modules). Now assuming a separate cockpit data bus (see Figure 10), and for illustrative purposes another processor, the crew interface software from the various subsets can now be mapped into this processor at the time of total system integration.

A structure has now been created which allows the various functional areas to develop hardware and software independently, albeit under certain constraints, yet also prepares for total system integration. Briefly stated, the constraints would be: to write all system level software in an HOL in accordance with a priori established rules, eliminate system level processing in the various sensors, and to use generic control/display hardware during development that is a functional subset of this total cockpit. Further, bus concepts that are evolving would treat the data bus interface as any other interface tied to the multiprocessor global bus thereby making the data bus transparent to future high speed data bus standards. As shown in Figure 11, the various data buses would be geographically distributed on an airframe at the time of total system integration even though development of the subsystems used functional distributions. Finally the multiprocessor would exhibit certain characteristics such as a global memory tied to the global bus, local memory with each microprocessor, and a very high speed interface by which a global bus in one multiprocessor can be tied to the global bus of another multiprocessor which is executing other functions or possibly the same functions because of fault tolerant considerations.

4.4 A Few Years Hence

The multiprocessor/multibus architecture evolved is in essence technology transparent; however, a few points must be made with regard to further evolution that includes hardware technology (such as VHSIC chips) and software technology (such as sensor fusion algorithms). Assuming that some of these processing functions will be located in special modules (e.g., signal processors) also tied to the processor global bus, it becomes necessary to create a means whereby the very wide bandwidth data from the sensors can be fed to these modules. This can be achieved via direct ports to the multiprocessor or if fault tolerant designs are to be achieved a very high speed sensor data bus may have to be synthesized. Much thought is currently being focused on this area and a number of concepts are evolving. The U.S. Army Avionics Laboratory is pursuing a number of studies to determine which of the concepts are applicable to helicopter systems.

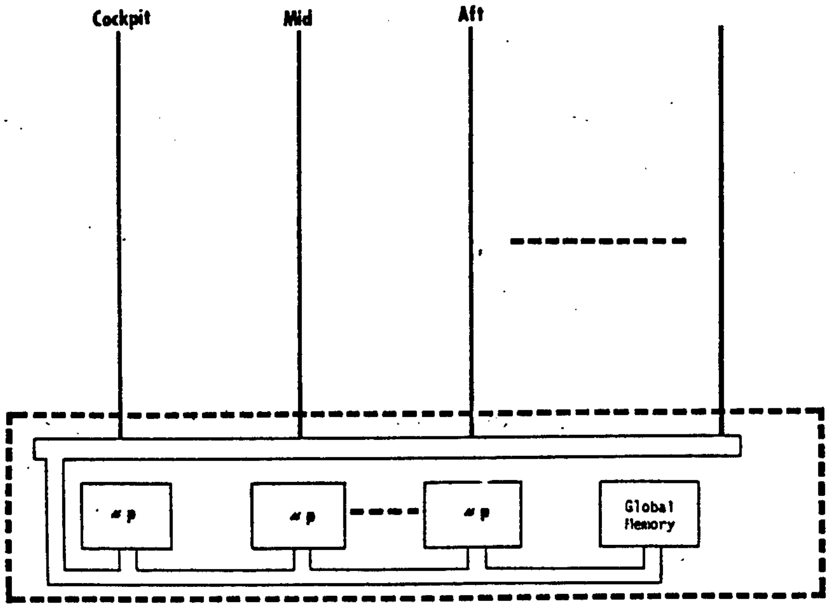


Figure 11. Multi-Processor/Multi-Bus Architecture

4.5 Conclusion

The concepts presented in this paper represent an attempt to arrive at an architecture which fits the avionics technology available today and that which will be available in the near future. Current efforts to implement this technology in the U.S. Army Avionics Laboratory are using currently available microprocessors configured as a multiprocessor and an available Ada compiler. A procurement strategy has evolved in parallel with this architecture that uses form, fit, function specifications and interface control documents so as not to restrict future procurements to today's technology. Certainly much work remains to be accomplished to evolve the concept presented here; however, it is fully expected that over the next several years, enough experience will be gained to achieve an architecture that will meet the needs of the demanding helicopter missions of the future.

4.6 Reference

Dasaro, J.A., Elliott, C. T., "Integration of Controls and Displays in U.S. Army Helicopter Cockpits," Proceedings of 32nd NATO Guidance and Control Panel Symposium on The Impact of New Guidance and Control Systems on Military Aircraft Cockpit Design, Stuttgart, Bad-Cannstatt, Germany, May 1981.

This Page Intentionally Left Blank

Chapter 5

THE F-16 AVIONICS MULTIPLEX SYSTEM

5.1 F-16 MULTIPLEX SYSTEM

The F-16 development program coincided closely with the initial publication of 1553(USAF) and the F-16 then became the first vehicle to use and flight-test a 1553(USAF) compatible multiplex data bus system.

The F-16 data bus system is characterized by an extremely simple approach to architecture, bus control, redundancy management, mechanization, and bus control transition technique.

5.1.1 Application Area

The F-16 data bus is basically limited to the avionic system (AMUX) with essentially all major avionic subsystems using the bus for data transfer. In fact, the only major subsystem absent is the flight control system.

Nine different avionic subsystems interface directly to the F-16 data bus:

a. Fire control computer (FCC)
b. Fire control and navigation panel (FCNP)
c. Inertial navigation unit (INU)
d. Fire control radar (FCR)
e. Radar electro-optical display (REO)
f. Central air data computer (CADC)
g. Head-up display (HUD)
h. Stores management set (SMS)
i. Target identification set, laser (TISL)

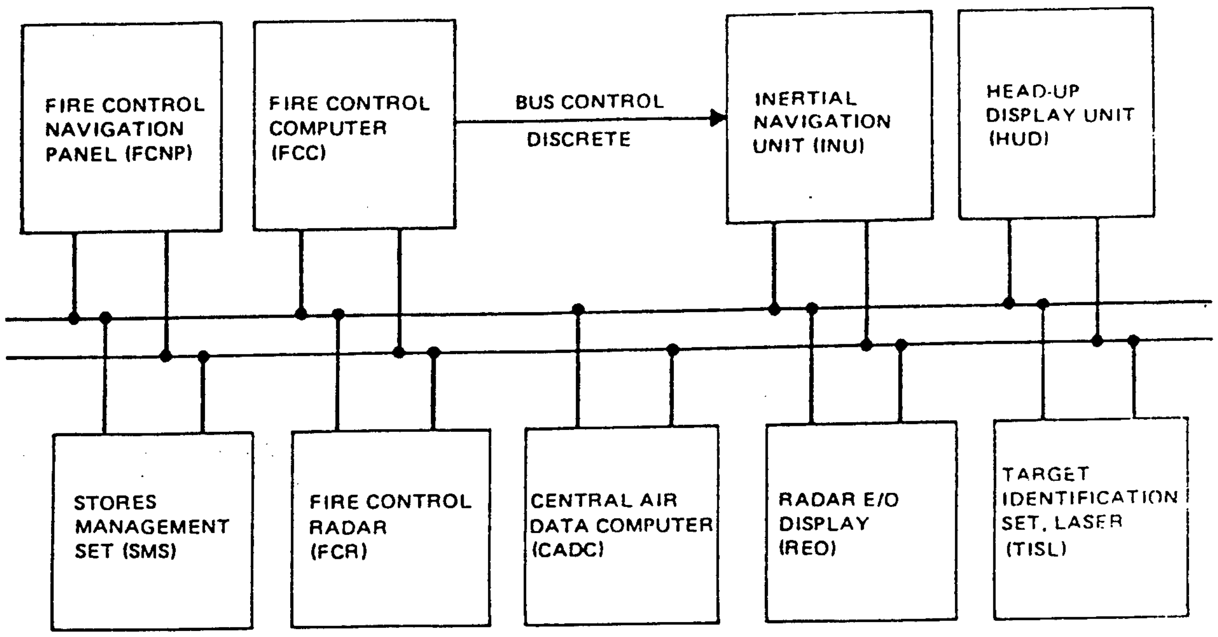

FIRE CONTROL NAVIGATION PANEL (FCNP)
FIRE CONTROL COMPUTER (FCC)
BUS CONTROL
DISCRETE
INERTIAL NAVIGATION UNIT (INU)
HEAD-UP DISPLAY UNIT (HUD)
STORES MANAGEMENT SET (SMS)
FIRE CONTROL RADAR (FCR)
CENTRAL AIR DATA COMPUTER (CADC)
RADAR E/O DISPLAY (REO)
TARGET IDENTIFICATION SET, LASER (TISL)

Figure 5-1. F-16 Multiplex System Architecture

All electronics required to interface with the multiplex bus are contained within the respective subsystem, thus completely eliminating the need for stand alone remote terminals (RT) or external signal conditioners. Thus, each subsystem provides data in digital form to the bus interface internal to the system, the only external signal interface being the serial multiplex bus.

5.1.2 System Architecture

Physical architecture of the F-16 avionic data bus system is shown in figure 5-1. A dual redundant bus network is used primarily to prevent a single bus fault (cable or connector) from rendering the system inoperative. With the exception of the SMS, none of the subsystems has functional redundancy. The SMS has two identical computers housed in one line replaceable unit (LRU). Each computer is connected to one of the 1553 buses through its own 1553 interface.

The F-16 data bus system was designed to be compatible with the interface requirements of 1553(USAF). The F-16 data bus system contains few actual deviations from 1553(USAF). Most would be more accurately classified as clarifications or additions to the original (no revision) standard.

5.1.2.3 Multiplex Cable Assembly

The F-16 data bus uses a very short cable assembly. Although 1553(USAF) allows up to 300 ft of cable, the F-16 main bus is only 30 ft long. All subsystems are attached to the bus by stubs which are connected to the main bus by transformer-resistor coupling networks. Except for provisions for the TISL system, the stubs vary in length from approximately 2.5 ft to 16.7 ft. The TISL provision includes a 22.5 ft stub to an externally mounted PAVEPENNY pod.

5.1.2.4 Bus Protocol

The functional architecture of the F-16 multiplex data bus system is shown in figure 5-2. All transactions are command/response with bus control centralized in the FCC. A backup bus control capability resides in the INU. Controller-to-terminal, terminal-to-controller, and terminal-to-terminal exchanges, as defined by 1553, are used. Terminal addresses are hard wired within the remote terminals. Any subsystem is capable of receiving a command on either bus at any time. A subsystem always acts on the latest command word received. If a second command word is received (on either bus) while a previous message is being received, the subsystem interrupts receipt of the first message and accepts the latest command. This feature also allows a transmission on one bus to be interrupted by a subsequent command on the second bus.

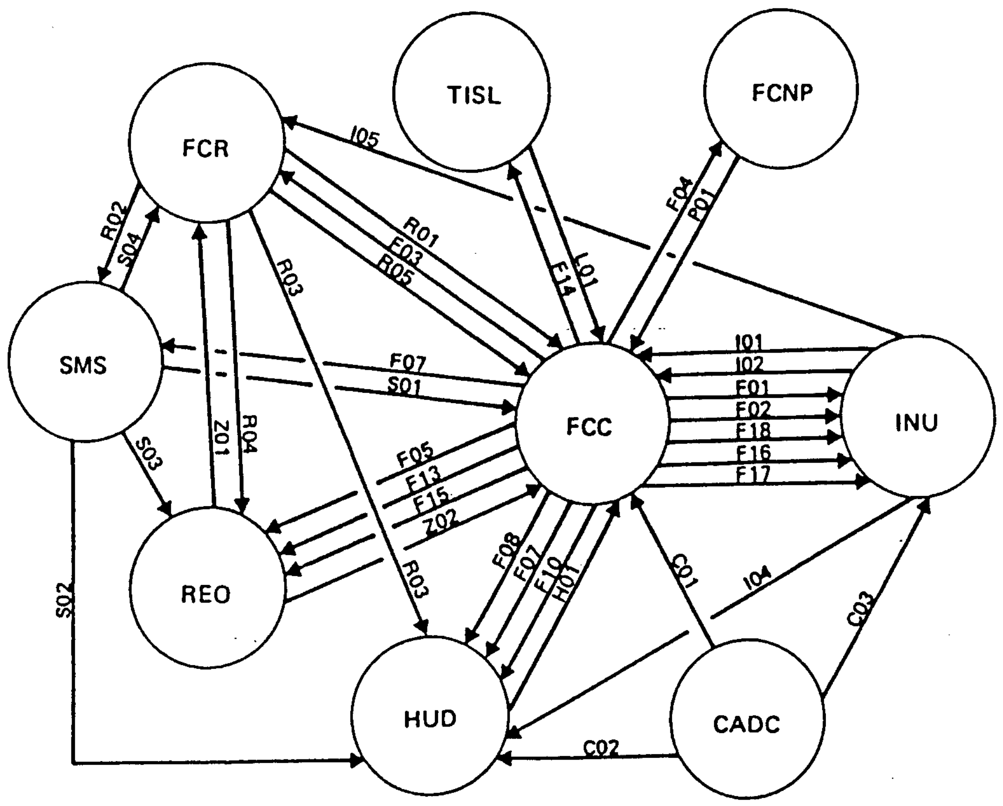


Figure 5-2. F-16 Multiplex Data Bus functional Block Diagram (Primary Mode)

All bus transactions are strictly scheduled by the FCC. Interrupt servicing as such is not allowed, but special servicing may be requested by a status word during a regular transaction.

Invalid Manchester word synchronization is used, with all subsystems using individual asynchronous clocks. A degree of synchronous operation is also possible as the central controller has the capability to command all terminal clocks to reset simultaneously. However this capability is not presently used. Instead, time-critical data, such as inertial platform measurements, is transmitted with a time tag so that individual users may establish latency of this data.

The use of time tag or algorithmic compensation has proved to be entirely satisfactory in removing the few variable latency problems encountered, thereby preserving the inherent simplicity of asynchronous operation.

Messages are transmitted in blocks ranging from 1 to 32 data words. The basic message transfer rate is 50 Hz with slower rates available in powers of two submultiples. The lowest data rate is 1.5625 Hz. Therefore, the major frame rate is 640 ms.

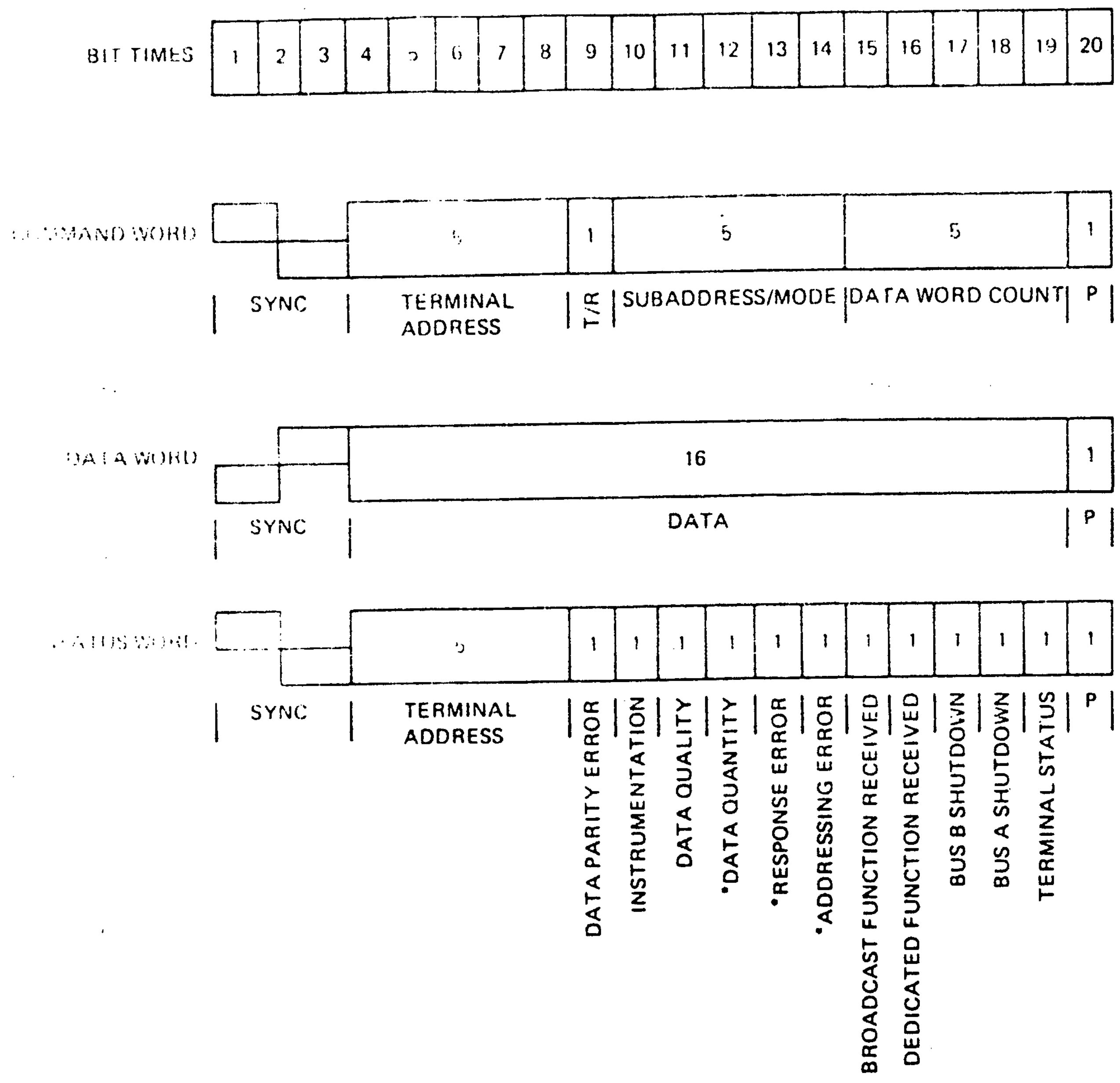
Command, status, and data word formats are as shown in figure 5-3. The command word is composed of a sync waveform, subsystem address, transmit/receive bit, subaddress/mode field, data word count, and a parity bit. Status word bit assignments are as shown. The data quantity, response error and addressing error bits (designated by * in fig. 5-3) are always transmitted as 0's on the bus. The bits may be set in the status word by the bus controller (FCC) after the word is received as an internal record of detected message completion failures in RT-to controller transfer.

The subaddress/mode field in the command word is used to indicate subaddress or function commands per 1553. Subaddresses are used to identify specific data blocks to be transferred. Function commands are indicated when the subaddress/mode field contains all logic 1's.

Only two function commands (mode codes) are used by the F-16. These assignments are shown in figure 5-4. The transmit status command (00000) causes the subsystem to reset and initialize its receiver logic and respond with its status word only. In addition, if a subsystem receives a function command (mode code) with any bit pattern other than 00000 or 00001, the transmit status command is assumed.

5.1.3 System Control

The F-16 multiplex data bus system uses a simple control philosophy. The FCC, when operating, acts as the bus controller. If the FCC is not operating, the INU assumes bus control. This concept is further simplified by the restriction that the FCC can never operate as a remote terminal.



*Used only by FCC for internal status information.

Figure 5-3. Word Formats

Command interpretation	Word count field (Bit time)				
	15	16	17	18	19
Transmit status	0	0	0	0	0
Reset timer	0	0	0	0	1

Note: Any other bit pattern will cause RT to "transmit status."

Figure 5-4. F-16 Function Word Commands (Mode Codes)

5.1.3.1 Fault Isolation and Redundancy Management

All bus control is based on the ability to communicate. Communication status assessment is established through periodic polling of each terminal. Polling occurs at the basic frame rate of 1.5625 Hz. Based on the polling results data transfer with a subsystem is either established or deleted. If a subsystem responds to a poll, data communications are established. If the subsystem fails to respond for two consecutive polling periods, that subsystem's data transfer commands are deleted from the controller's current command table. Thus, periodic polling allows a subsystem's communication status changes to be discerned without reference to any additional input.

The ability-to-communicate approach also results in one of the simplest strategies possible for selection of the redundant channel. The controller simply always uses the channel that worked last. For example, a successful transfer on bus A would result in the next transfer of the same block also being attempted on bus A; however, if the first attempt on bus A failed, then the retry of that transmission would be attempted on bus B. Thus, communications will continue on the channel that is functioning. Note that the retry is limited to once per scan of the command table and is always initiated on the alternate bus. If the retry fails, the command is skipped. The sequence is "fail once, retry on alternate; fail twice, go to next command."

5.1.3.2 Primary and Backup Control

The F-16 FCC normally functions as bus controller. In case of an FCC power down or self-test failure, bus control is assumed by the INU. Once again, the simplicity of the F-16 AMUX system is apparent in the mechanization of the bus control transition technique. If the FCC is powered down or becomes inoperative, bus control is passed to the INU by a single discrete between the two units. During operation, the INU periodically samples the discrete for a high-voltage or high-impedance condition. If two consecutive samples are in the pass control or NO-GO state, the INU assumes bus control responsibilities. The INU then continues periodic sampling of the discrete and relinquishes control to the FCC immediately upon detection of a low-voltage GO condition on the discrete.

Bus control is greatly simplified in the backup mode because the FCC and TISL are completely eliminated from the system. This is apparent by comparison of the backup functional interface shown in figure 5-5 with that of the primary mode (fig. 5-2).

5.1.4 Bus Controller

The FCC's prominence in the primary data flow pattern (see fig. 5-2) figured heavily in the selection of the FCC as the primary bus controller. Also, General Dynamics was responsible for developing the operational software for the FCC, thus ensuring that the prime contractor maintained responsibility for system integration.

Again referring to figure 5-2 and following the previously established line of reasoning, it became apparent that the INU would figure most prominently in the data flow pattern should the FCC fail. Therefore, the INU was selected to perform the backup bus control function.

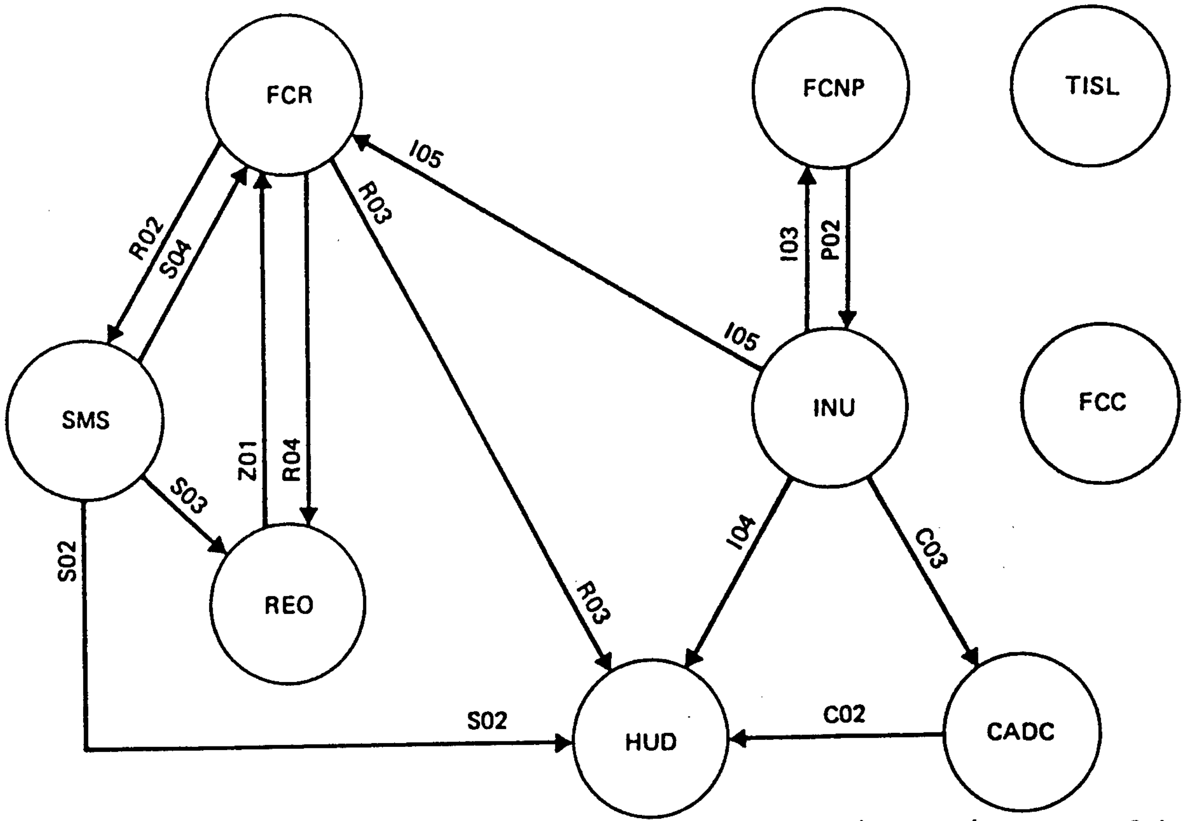


Figure 5-5. F-16 Multiplex Data Bus Functional Block Diagram (Backup Mode)

5.1.4.1 Primary Bus Control Implementation

Primary bus control resides in the FCC. The FCC is a Delco M362F computer procured for the F-16 and programmed by General Dynamics. Actual bus control is maintained by a microprogrammable hardware controller that is initiated periodically by the FCC operational flight program (OFP). This controller, called the serial data interface (SDI) reads and executes bus transfer sequences stored in the FCC main memory. Once initiated by the FCC OFP software, the SDI will continue to read and execute the command table until either (1) the command sequence is complete or (2) a transmission fails to complete successfully. Software intervention is required only to start the SDI processor, analyze poll responses, and to process SDI-initiated interrupts such as "command failure" or "command task complete." A command failure interrupt is generated to the CPU when any commanded bus transmission fails to complete successfully. A command task complete interrupt is used twice per minor frame: (1) to inform the OFP that key input data has been received and (2) to indicate that all scheduled transfers for the current time frame have been completed. The first interrupt may be generated after any designated SDI command.

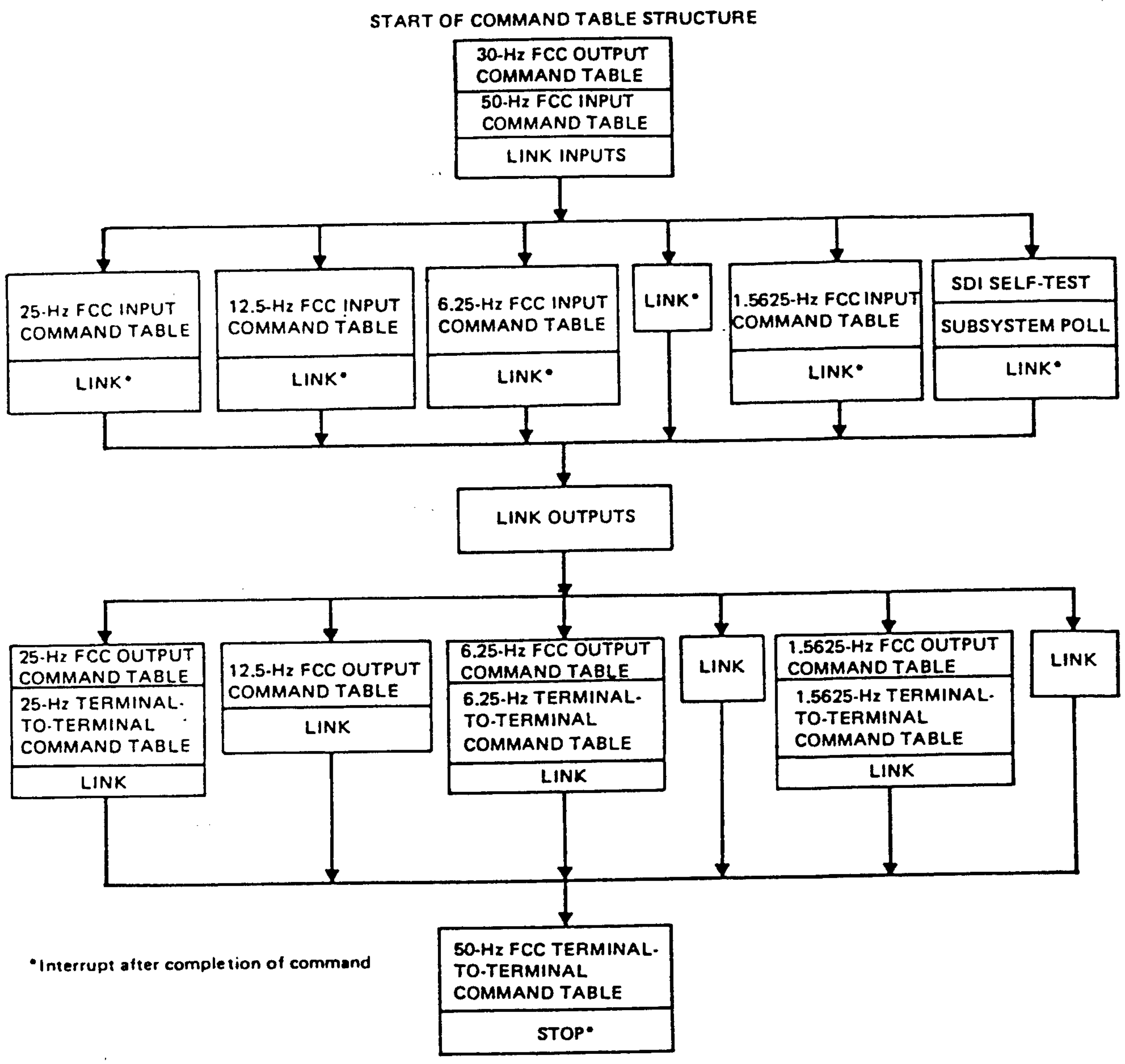

START OF COMMAND TABLE STRUCTURE
30-Hz FCC OUTPUT COMMAND TABLE
50-Hz FCC INPUT COMMAND TABLE
LINK INPUTS
25-Hz FCC INPUT COMMAND TABLE
LINK*
12.5-Hz FCC INPUT COMMAND TABLE
LINK*
6.25-Hz FCC INPUT COMMAND TABLE
LINK*
LINK*
1.5625-Hz FCC INPUT COMMAND TABLE
LINK*
SDI SELF-TEST
SUBSYSTEM POLL
LINK*
LINK OUTPUTS
25-Hz FCC OUTPUT COMMAND TABLE
25-Hz TERMINAL-TO-TERMINAL COMMAND TABLE
LINK
12.5-Hz FCC OUTPUT COMMAND TABLE
LINK
6.25-Hz FCC OUTPUT COMMAND TABLE
6.25-Hz TERMINAL-TO-TERMINAL COMMAND TABLE
LINK
LINK
1.5625-Hz FCC OUTPUT COMMAND TABLE
1.5625-Hz TERMINAL-TO-TERMINAL COMMAND TABLE
LINK
LINK
*Interrupt after completion of command
50-Hz FCC TERMINAL-TO-TERMINAL COMMAND TABLE
STOP*

Figure 5-6. Bus Control Command Table Structure

In addition to data transfers, the SDI may also be commanded (by the OFP software) to perform various internal functions such as command sequence branches, internal self-test, or SDI stop.

The OFP controls data transfers using a time-slice executive structure operating at a 50-Hz maximum computational rate. The SDI thus is commanded to initiate a transfer sequence once per 20 ms minor frame. Actual avionic interface data are structured into blocks with fixed transfer rates that are powers of two submultiples of the basic 50-Hz rate. Therefore, the command sequence in each minor frame will consist of data blocks from the 50-Hz group plus blocks from one of the submultiple groups. Because the lowest transfer rate is 1.5625 Hz, 640 milliseconds are required to complete a full data transfer sequence (major frame).

This periodic transfer sequence is implemented by linking the SDI command table prior to each minor frame. This is accomplished by setting two link commands to provide the desired path through the command table (fig. 5-6). The content of the command table is modified by the results of poll processing to eliminate the transfer commands of those subsystems that are not actively communicating.

Polling is accomplished at the major frame rate of 1.5625 Hz to ensure that satisfactory communication can be established with a terminal before a data transfer is attempted. Polling is accomplished using the F-16 dedicated function (mode code) command, which requests the addressed terminal to respond with its current status word only. The dedicated function command is distinguished by all "1's" in the subaddress/mode field.

Any poll command that fails (whether because of a reported fault status or a transmission failure) causes a CPU interrupt and subsequent software recording of the poll command failure. The result of each subsystem poll is then masked with the results of the previous poll and used to delete the data transfer for that subsystem from the command table if two successive poll failures are recorded. The next successful poll response from that subsystem will immediately reinstate the transfer command and so reestablish communication with the polled subsystem.

In addition to poll response errors, data transmission errors also are handled by special error-handling interrupt software. The error-handling software indexes an error response table that determines the appropriate error response for each command. The error response table will indicate (1) whether the command is to be retried, (2) the bus to be used for the retry, and (3) whether the transmitted data (if any) should be invalidated. As previously noted, all retries are currently initiated on the alternate bus, but sufficient software flexibility exists to allow the retry mode to be selectively changed on an individual command basis if so desired. The FCC used 950 16-bit words of memory for bus control. This includes the complete control algorithm for (1) SDI start commands, (2) interrupt handler for command link selection, (3) error interrupt handler, (4) poll analysis module, and (5) bus command tables. CPU processing time comprises less than 2% of the machine duty cycle.

5.1.4.2 Secondary Bus Control Implementation

Secondary or backup bus control is provided by the INU. The INU periodically samples the bus control discrete from the FCC and assumes bus control after two consecutive NO-GO samples. The basic bus control concept used by the INU is essentially the same as that previously discussed for the primary controller. The backup control algorithm, however, is considerably simpler than that of the primary system for two reasons. First, the number of data blocks to be managed is much smaller. Second, fault reporting and error recovery requirements are considerably reduced. The INU contains a hardware DMA controller that is commanded by the INU OFP software in the same manner as the FCC. The only difference in implementation is that backup redundancy management is hardware controlled by the INU, whereas OFP software in the FCC controls redundancy management in the primary bus control mode.

5.1.5 Remote Terminal

The F-16 multiplex data bus system interfaces with and provides complete communication with nine major subsystems, as listed in section 5.1.1. All bus interfaces are integral to the subsystems they serve. This approach drastically reduces integration problems associated with standalone RT/signal conditioning systems.

Of the nine subsystems, seven always act as RT's only. One, the FCC, acts as the primary bus controller and is deleted from system communication in the backup mode. It can never act as a RT. The INU can act as either a RT (in the primary mode) or a bus controller (in the secondary mode).

5.1.5.1 Subsystem Interface

Because all bus interfaces are integral to the subsystems that they serve, the usual subsystem interface is solely the responsibility of the avionic's supplier and, in fact, does not exist external to the subsystem. Thus, the communication interface with the subsystem is limited to the 1553 bus. None of the F-16 subsystems use a standard interface module. The bus interfaces within the various subsystems represent independent designs by six different suppliers.

5.1.5.2 Fault Isolation and Redundancy

Although the F-16 has a dual redundant bus network, only the stores management set is fully dual redundant. None of the other subsystems have functional redundancy. The SMS utilizes two identical AMUX interface modules. Each AMUX module serves one redundant half of the SMS and communicates with one of the two data buses.

Fault conditions within a subsystem which could affect data validity are "OR'ed" into a single terminal status bit which is returned in the status word. Fault conditions included in the terminal status bit are determined on an individual subsystem basis. The basic ground rule, however, is that to be included in the terminal status bit, the failure must affect validity of all data transmitted by the subsystem. Other fault conditions are detected by the system controller based on ability to communicate.

A 1553 feature that is implemented in each F-16 terminal is "operation on latest command word." This feature requires that a terminal act on the latest command word received on either bus even though it may interrupt receipt or transmission of a message in progress. A message being received may be interrupted on either bus. A message being transmitted may be interrupted by a command on the alternate bus. This is the only condition in the F-16 system under which both buses may be in use simultaneously. This feature is potentially useful as a priority override or to shut down a faulty transmitter.

A terminal status word only may be requested from an individual terminal by use of the dedicated function command designated by an all 1's subaddress/mode code field, as described in section 6.1.2.4.

Section II

System Test Activities

This Page Intentionally Left Blank

Chapter 6

MULTIPLEX SYSTEM TEST

Tests of 1553 aircraft data bus system will take place (1) at the facilities of suppliers of LRU's with 1553 interfaces, (2) at a "hot bench" generally located at the airplane company, and (3) during flight test. Flight tests will usually be conducted near the airplane company and possibly at military bases and other locations by the eventual military using command. the purpose of this section is to briefly describe hardware, test procedures, and test philosophy of the various levels of testing that have been found useful for test of 1553 data bus systems.

6.1 Scope of Tests

Data bus interfaces are not usually designed, developed, and tested independently of all other LRU interfaces. Although the discussion centers on tests of subsystems with 1553 interfaces and system test, it should be understood that many other design and test activities are required to successfully complete avionics integration for an airplane. The 1553 terminal design and test form a part of these activities. This is so because 1553 defines a terminal as: "The electronic module necessary to interface the data bus with the subsystem and the subsystem with the data bus." Tests of subsystems connected to the 1553 data bus usually include verification of interface functions on the subsystem side as well as the bus side of the terminal.

Design verification tests of 1553 terminals are an important part of system and subsystem design. Table 6-1 is a summary of design procedures. Table 6-2 is a summary of test procedure development. Each table is divided into two parts, procedures for the subsystem side as well as the bus side of the terminal. The procedures presented in the tables should be used to scope the test activities.

Table 6-1. Design Procedures Guidelines

1. Design procedures for avionic subsystem interface to remote terminal (RT):
   a. The Government or the integration contractor will provide the standard interface modules' (IM) definition to the avionic contractor, when standard IM's are used.
   b. Contractor will develop the electrical interface definition between the avionic subsystem and the RT IM.
   c. Perform an engineering design of the electrical interface.
   d. Manufacture the breadboard to accommodate a standard IM.
   e. Develop support test equipment to generate the IM interface input and output signals.
   f. Define acceptance test.
   g. Perform acceptance test.
   h. Evaluate test results.
   i. Report out-of-limits conditions for failed test.
   j. Apply test results to reevaluate and improve interface performance.

2. Design procedures for avionic subsystem interface to MIL-STD-1553 bus:
   a. Evaluate interface definition of chosen avionic subsystem.
   b. Design and develop the electrical interface definition between the avionic subsystem and the 1553 bus.
   c. Design a bus interface unit and subsystem interface.
   d. Develop support test equipment to generate the subsystem and bus input and output signals.
   e. Define acceptance test.
   f. Perform acceptance test.
   g. Evaluate test results.
   h. Report out-of-limits conditions for failed test.
   i. Apply test results to reevaluate and redesign interface performance.

6.2 Typical 1553 Bus Checkout Systems

The purpose of this section is to introduce the types of bus checkout systems that are frequently used. Generally, there are three classes:

a. The bench or suitcase 1553 multiplex bus tester
b. The entire avionics hot bench
c. The programmed bus monitor for flight test

The bench or suitcase testers may also be used to support troubleshooting during hot bench and flight tests.

Table 6-2. Test Procedures Guidelines

1. Methods for developing test procedure for remote terminal (RT) to avionic subsystem interface:
   a. Coordinate with interface equipment designers.
   b. Identify subsystem interface requirements.
   c. Determine nominal interface operation.
   d. Identify error modes and off-nominal operations.
   e. Determine desired form of test results.
   f. Determine test equipment requirements.
   g. Establish system time lines within protocol constraints.
   h. Flow chart test requirements.
   i. Recommend test procedures.

2. Method for developing test procedure for a MIL-STD-1553 to avionic subsystem interface:
   a. Coordinate with interface equipment designers.
   b. Identify subsystem interface requirements.
   c. Determine nominal interface requirements.
   d. Identify error modes and off-nominal operation.
   e. Determine desired form of test results.
   f. Determine test equipment requirements.
   g. Establish system time lines within MIL-STD-1553 protocol constraints.
   h. Determine MIL-STD-1553 parameters that need testing.
   i. Recommend test procedures.

6.2.1 Multiplex Bus Tester and Simulator

A simulator is a versatile data bus test instrument compatible with MIL-STD-1553 for applications in the engineering laboratory, in system integration laboratories, and as a portable instrument for fault isolation.

Simulators have full capability to act as a bus controller, both sending and receiving data bus messages. The simulator will transmit a command word and a selected number of 16-bit data words. The command word is front panel selectable, and the 16-bit field of each data word is loaded into memory from the front panel switches. The proper polarity sync patterns and parity bits are added to each word to provide the correct word formats.

Simulators also have the features necessary to receive the message from a remote terminal, corresponding to the command word address that was transmitted. The unit will display the status and all data words as selected from the front panel control switches.

Typically simulators have the capability to receive and verify valid and invalid test words and messages.

6.2.2 Avionics "Hot Bench"

Hot bench is a commonly used term to describe a system development laboratory (SDL) or system integration laboratory (SIL). Such SIL's or SDL's provide simulation capability and data recording capability that are used in the development of avionic hardware and software. To allow for incremental testing of avionics interfaces, simulators of airplane subsystems such as the radar or stores are used. The simulators provide realistic inputs and responses so that dynamic conditions may be evaluated in the laboratory. This is in contrast to the capability of bench or suitcase testers, which usually can evaluate only a command and a response. The simulators in hot benches are a substitute for unavailable hardware, and an intermixing of prototype or production airplane hardware with simulators is usually possible. By this means, airplane hardware can be incrementally added to an avionic system. Whenever the interface is the 1553 data bus, rapid resubstitution of the simulator for the airplane hardware permits the isolation of problems or anomalies.

Several benefits of integration with 1553 data buses become apparent during system test. The data bus approach requires integrating only one electrical interface per subsystem versus multiple interfaces in the point-to-point method. The single interface also allows more of the integration activity to be done at the subsystem level using one special test fixture, which might be too costly with many unique point-to-point interfaces. Simulating the data bus interface of subsystems can easily be done using a computer with a data bus interface as the simulator. The equivalent simulation in a point-to-point architecture may require several special-purpose interfaces to be developed. The data bus will also accommodate unexpected integration problems such as added data words, changes in update rates, and rerouting of data parameters.

6.2.3 Bus Monitor and Airborne Instrumentation

System designers should make provision for the connection of a bus monitor and avionics instrumentation capability. Provision will usually be a stub or connection properly terminated when not in use on prototype and test airplanes. With this connection available, a bus monitor may be used during flight test to acquire selected bus messages. Recall that MIL-STD-1553B describes bus monitor operation in the following way:

> A terminal operating as a bus monitor shall receive bus traffic and extract selected information. While operating as a bus monitor, the terminal shall not respond to any message except one containing its own unique address if one is assigned. All information obtained while acting as a bus monitor shall be strictly used for off-line applications (e.g., flight test recording, maintenance recording or mission analysis) or to provide the back-up bus controller sufficient information to take over as the bus controller.

Bus monitors are usually implemented using a digital computer, appropriate memory for buffering, and magnetic tape for recording. Several suppliers of bus monitors and airborne instrumentation are qualified for flight test.

This Page Intentionally Left Blank

Chapter 7

F-16 C & D TESTING

7.1 Background

In 1979, early in the production run of the F-16A/B, the United States Air Force defined a program for the continued expansion of F-16 capabilities. In particular, the program was intended to enable the F-16 to perform both the night, low-level, ground attack mission, and the all-weather, multi-target, air-to-air mission. To accomplish the first task, the future F-16 would utilize the Low Altitude Navigation and Targeting Infrared for Night (LANTIRN) system which incorporates navigation and targeting forward looking infrared sensors and a terrain following radar. An improved fire control radar would enhance both the air-to-ground and the air-to-air capabilities of the F-16, and Advanced Medium Range Air-to-Air Missile (AMRAAM) would be the beyond-visual-range weapon. When formalized, this enhancement plan was defined as the three step F-16 Multinational Staged Improvement Program (MSIP).

Stage I of F-16 MSIP commenced in mid-1981. All F-16A/B airplanes produced since that time have structural and wiring provisions for the later retrofit of advanced systems. These modifications include inlet hard points for sensor pods and wiring to wing stations for future weapons.

MSIP Stage II is the F-16C/D airplane which includes the following major changes from the F-16A/B:

---

by Kevin Dwyer, General Dynamics, Fort Worth Division and Lt Col Tom Meschko, F-16 CTF, Edwards AFB CA. This text originally appeared in the 1984 Report to the Aerospace Profession, XXVIII Symposium Proceedings, Society of Experimental Test Pilots, 1985.

a. Improved Cockpit.

(1) Upfront controls and display for communication, navigation, IFF, and fire control computer interface,

(2) Two CRT multifunction displays (MFD) for control and display of sensors, weapons, and stores management,

(3) Wide field-of-view head-up display (WFOV HUD),

(4) Data transfer equipment which lets the pilot "load" his mission into the airplane via a solid state cartridge;

b. New/Improved Avionics.

(1) APG-68 Radar increases detection and acquisition ranges, and adds Track-While-Scan and additional Air-to-Ground modes,

(2) Combined Altitude Radar Altimeter (CARA),

(3) Advanced Stores Management Set

(4) Expanded Memory/Speed Fire Control Computer;

c. Airframe Changes.

(1) Structural provisions for the Airborne Self-Protection Jammer (ASPJ) system, including an enlarged vertical tail fairing,

(2) Main generator upgrade from a 40 KVA generator (on F-16A/B airplanes) to 60 KVA, and the addition of a 10 KVA standby generator (the other features of the F-16A electrical system have been retained),

(3) Increased capacity environmental control system to allow all the avionics (and the pilot) to keep cool,

(4) New main gear tires to permit an increase of maximum takeoff gross weight to 37,500 lbs.,

(5) Revised leading edge flap schedule to reduce hinge moments during maneuvering,

(6) Expansion of 9g maneuver envelope to 26,100 lbs. gross weight (versus 24,100 lbs. for the F-16A/B);

d. Added Weapons Carriage.

(1) Necessary interface for employment of the Imaging Infrared Maverick (AGM-65D),

(2) Partial provisions for AMRAAM and future weapons.

Stage III of the F-16 MSIP will be the step-by-step integration of future systems as they become available.

7.2 Avionics Testing

The MSIP II avionics tests are presented in the context of software development "facts of life" and the methods used by the F-16 Combined Test Force at Edwards AFB.

Compared to the F-16A/B, the MSIP II, or F-16C/D, software effort is massive, having several individual planned updates of over 20,000 words which is on the order of the total number of words in all of the updates in the F-16A/B. It is undeniable that software takes time. There is significant lead time required with designing and programming software just as lead time is required for hardware development. For example, the lead time required for one thousand words of software for the four major computers in the core of the F-16C avionics ranges from 9 months for the easiest one, the multifunction display system, to 12 months for the expanded fire control computer.

7.2.1 Development and Flight Test Cycle

There are four major steps in the overall block release process: design, programming, integration, and then flight test. The total time required from the beginning of design until the final product comes out of flight test is two to two-and-one-half years. Since the four steps are sequential, personnel and resources used in one step are relatively free to begin work on the next block when they finish their major effort on the current block. Such overlapping permits blocks to be generated approximately twice as fast as a strict serial process would allow. Unfortunately, however, the overlapping cannot reduce the time to effect major changes resulting from flight test. Major block 1 flight test findings cannot be put into the immediately subsequent block because the design phase of that block is complete before the flight testing of the previous block is accomplished.

Consequently, all the findings from block 1 go into block 3. Block 2 is almost an independent effort, though parallel and staggered a year after block 1. The major flight test generated changes from block 2 will not be incorporated until block 4. The significant point is that in a large program major updates and flight test findings cannot go into production until two years downstream.

It is important to understand the flight test cycle inherent in this process. When a "finished" tape completes system integration in the laboratory, it is released for flight test. In flight test, the tape must sequentially undergo three basic phases of testing. The first is pure development where it is determined whether or not the tape simply functions as designed; it does not address the merits of the design, only that everything that is supposed to be is there and operates as designed. If any part or feature is/does not, a potential change candidate is generated; essentially, it is back to the drawing boards for correction.

Once a tape gets through the first phase, it advances to developmental test and evaluation (DT&E). In this phase we test and evaluate the design, its overall performance, and its integration, not only from a system point of view, but also from a pilot/vehicle interface point of view. It is also in this phase that full operability and initial operational assessments are made as well as testing for specifications compliance. Like features which do not function as designed in the pure development loop, deficiencies in design or specification compliance also generate potential change candidates.

Once a tape completes the DT&E phase, it advances to the third loop, the operational test and evaluation phase; it is in this phase that the system is evaluated and tested to determine how well it actually meets overall mission requirements and does the job operationally. Again, when identified, design and performance deficiencies become potential change candidates. When a tape makes it through all three phases, it is finally released for production. As with hardware flight testing, all of the testing conducted in these three phases is governed by test plans. Beyond that, however, the processes become considerably less parallel. There are several pitfalls in the flight test cycle inherent to software testing, which are not generally found in hardware testing.

7.2.2 Management of Configuration

As mentioned above, each phase generates potential change candidates. These potential change candidates must be properly managed. Otherwise, there is real potential that items singularly identified in flight test may be incorporated because they are easy or appealing, but not necessarily the consensus of the entire participating test community. Corrective effort may be expended without much visibility into its overall importance or its relationship to the "big picture." This may result in insufficient effort on more important items. There is also the risk of items being discarded or otherwise lost because of unilateral determination that they were not within scope. There is a tendency for only easy items to get fixed while the harder and maybe more important major items are deferred. And finally, there is the possibility that the items or inputs from the flight test might be misinter- preted and, therefore be "fixed" wrong. All of these problems result in valuable time lost, the possibility of the final configurations being degraded, and, at the very least, more cycles through the top loop. In order to effectively manage the change and fix effort, procedures to formalize, categorize, prioritize, and track all potential change candidates should be applied in the feedback portion of each loop. Once the change candidates find their ways into laboratory integrated, updated versions of the tape, another pitfall of software testing arises: inadequate configuration control.

The major problem with configuration control is insufficient documentation--not documenting and keeping track of what actually is in the new tape. Insufficient documentation confuses the results. And it is very easy to lose track of the actual configuration, especially if you've had several versions varying from prototype to production in five or more subsystems. If configuration is in question, results are invalid or inconclusive at best. Unless complete refly is accomplished, knowledge of the system is poor and possibly could result in a system with insufficiently tested or even partially untested portions being released to the field which could ultimately end up in costly retrofit. Strict documentation of all incorporated changes is required to maintain accurate configuration control.

7.2.2.1 Functional Tests

Because of the extreme complexity of today's advanced integrated software systems and the proliferation of intricate interfaces between the various subystems and modes, a fix or correction in one area frequently degrades another area. Unfortuantely, the integration laboratory does not always reveal all of the side effects. It is imperative, therefore, after each change or updated tape, that all of the functional tests (i.e. the top loop) be repeated to the functional tests (i.e. the top loop) be repeated to assure the desired fix(es) did not break or degrade other features or functions. In complex systems, the functionals may take several sorties which for unplanned updates are usually not provided for in the test plan nor is the requirement to repeat most if not all of the tests in the second and third phases.

Because of the extra requirements generated by unplanned updates, these should be kept to a minimum and should incorporate as many fixes at one time as practical. This requires implementing a control function between the ground integration phase and the first phase of the flight test cycle.

Unfortunately, the pitfalls inherent to change candidates tend to perpetuate the cycle, and in so doing generate yet another pitfall: the fly-fix-fly syndrome. Each unplanned update uses valuable time and in essence marks a new start. The end date or required delivery date seldom slips, however. This creates a compression of the available time in which to complete all the planned, required tests and invariably leads to an overwhelming sense of urgency and increased pressure to test the latest fix immediately, short-circuiting the configuration documentation and ignoring the consequences of the extra functional sorties. The risk of losing configuration control is increased along with increased likelihood of incomplete or incorrect fixes and therefore, even additional change candidates. Thus the compression increases with the need for repeating the cycle. Uncontrolled fly-fix-fly ultimately leads to significant overruns, far too much effort spent in the pure development loop, and too little effort spent on test and evaluation, which itself results in proportionately too much specification compliance at the expense of testing and evaluating the full operability aspects. This leads ultimately to having the true test and evaluation done by the user and, most likely, results in costly retrofit.

In order to preclude or minimize inadequate configuration control and limit the fly-fix-fly syndrome, a control function must be judiciously exercised to assure accurate knowledge of the configuration and to regulate the frequency of updates introduced into the first loop.

7.2.2.2 Simulation

An activity which offers potentially significant reduction in the number of required updates is developmental simulation. Simulation can optimize pilot interfaces and explore options which otherwise must eventually be done in flight or not at all. Simulation is also very important in minimizing surprises in flight test by reducing the number of fixes and configuration problems before flight test. This essentially minimizes the number of times that a tape has to go through the first loop. Simulation is needed not only early in the design phase, but also during flight test to look at the various design options for change candidates generated in any of the feedback loops from any of the flight test phases. A comprehensively constructed software flight test program might then include three flight phases governed by the test plans, with appropriate procedures to control and track the change candidates introduced or redesigned, with an effective mechnization to document and track the configuration as well as to control the fly-fix-fly tendency, and with adequate simulation for the initial and follow-on design efforts.

7.2.2.3 Watch Items List

In the F-16 test community, all of these measures have been instituted in varying degrees in the extensive F-16C/D software development program. The change candidates come from a carefully monitored "watch items" (WITS) list. To date we have generated over 600 different watch items, many of which have gone through the cycles and are now closed. The vehicle for turning worthy watch items into change candidates, for documenting and controlling configuration changes, and for judicious incorporation of updates is a formalized process which is implemented through structured weekly conference calls between the combined test force, the contractor(s), and the systems program office (SPO). During the weekly conference calls, the discussion focues on new WITS, ongoing contractor efforts from the previous weeks, and SPO implementation directives. This method has provided continuous tracking of all the change candidates, allowed visibility into the actual correctve efforts, placed proper focus and emphasis on the higher priority items, accurate tracking and control of configüration, and reasonable control of the fly-fix-fly tendency.

7.3 Conclusion

A significant tribute to the developmental feedback effectiveness of the F-16 software testing system was a cockpit remechanization which included not only software, but also hardware changes. The original software architecture sometimes required up to nine switch actuations to access some unforeseen, but as it turns out routinely required display and data formats. With mission requirements more clearly defined through flight test, and with redesign aided by simulation, a remechanization which reduced switch actuations for all data/display access to a maximum of two was implemented. This, in essence, became the major feature of Block 25B.

This Page Intentionally Left Blank

Chapter 7

INTEGRATION FACILITY FOR AVIONIC SYSTEMS TESTING "IFAST"

7.0 Abstract

This paper describes the Integration Facility for Avionic Systems Testing (IFAST) which is being developed on Edwards Air Force Base, California, to support avionics flight testing at the Air Force Flight Test Center (AFFTC). New avionics complexity has made effective avionics flight testing difficult, but success will continue at the AFFTC with support from the IFAST. IFAST provides a uniquely configured and sited building with test bays and automated systems. Common elements in test facilities and avionics standards make the IFAST approach of multi-user support technically and economically attractive. Air Force-contractor teams can apply "test-before-fly" techniques in IFAST which improve the in-flight utilization of test aircraft. High pay-offs can come from IFAST support applications like functional verification and familiarization, integrated troubleshooting, and resolution of relative performance issues which may arise when avionics modifications are made between flights. Past attempts to "fly quality into avionics" were very time consuming and expensive.

by James M. Underwood, Jr., IFAST Program Manager; Robert W. Bockstahler and Dennis H. Sheldon, VERAC, Inc.

7.1 Introduction

Since the IFAST is a major support facility for the AFFTC, an introduction to the IFAST should include some discussion of the applicable portions of the AFFTC's missions. The AFFTC is responsible to Air Force Systems Command for planning, conducting, and independently reporting on contractor/United States Air Force (USAF) development test and evaluation (DT&E) programs involving manned and unmanned aerospace vehicles and deceleration devices. Technical areas of responsiblity include:

(1) Performance and flying qualities.

(2) Structures, flutter, and climatic effects.

(3) Human factors and operational utility.

(4) Reliability, maintainability, and initial supportability.

(5) Compatibility of support and airborne equipment.

(6) Functional capability/compatibility of subsystems.

As a result of avionics proliferation in these aerospace vehicles and the increasing amount of software control over the avionics, more emphasis is required at the AFFTC in this last technical area on avionic subsystems. Furthermore, increasing costs and major changes in avionics technologies/applications made it prudent to develop the IFAST to insure that avionics DT&E at the AFFTC continues to be effective by producing the best avionics possible and reducing costs. The success of the IFAST in achieving these two primary goals is practically guaranteed by general agreement that:

(1) Engineering simulations have matured sufficiently to be used in lieu of actual flight testing in many important applications.

(2) Highly beneficial avionics ground testing can be done for one-hundredth the cost of flight testing (Ref. 1).

(3) Avionics ground testing in well equipped facilities can provide substantial benefits that real flying cannot (i.e., freeze, roll-back, and precise repeatability).

The IFAST development "got off the ground" in November 1982 when the building (Fig. 1) construction was completed. There are other primary efforts in the IFAST development which include the installation and demonstration of automated data processing systems (ADPS) and the phased adaptation of the basic capabilities to user programs. These development efforts are managed by the AFFTC's 6520th Test Group, but operations within individual IFAST program areas are managed by 6510th Test Wing combined test forces (CTFs) that are manned by program contractor and USAF personnel.

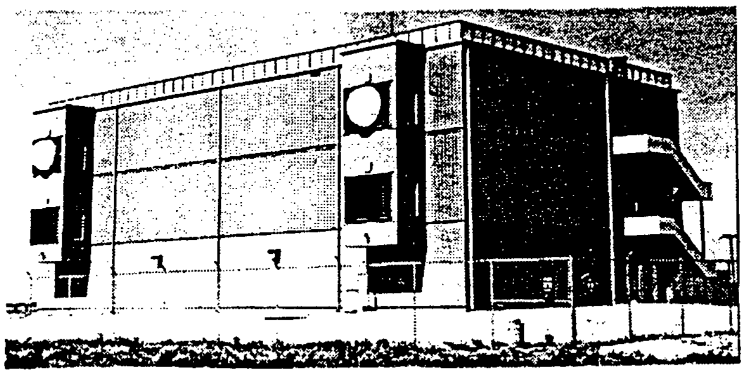


Figure 1. IFAST Building

Successful full-scale development (FSD) flight test programs involving complex avionics must have troubleshooting facilities like the IFAST at the flight test site. Any attempt to provide primary support from off-site facilities will introduce delays and monumental communications problems that will result is wasted time/money and inappropriate use of FSD aircraft to troubleshoot problems that could be resolved in ground test. These on-site facilities are not intended to replace the development/integration laboratories needed at contractor plants. A full flown avionics program like the F-15 or B-1B would keep both facilities very busy with different tasks.

7.2 Technology Impacts

Rapid Changes in avionics technology are forcing development of new test policies, methodologies, and resources. Important flight test elements like techniques, instrumentation, and data requirements have been impacted significantly. Consideration of the following functional changes provides better insight into these impacts:

(1) <u>Digital integration</u> of systems with <u>distributed processing</u>.

(2) Extensive capabilities under <u>programmable software</u> control.

(3) <u>New applications</u>.

7.2.1 Digital Integration

Utilization of the one megabit (MIL-STD-1553) and higher speed digital data buses in avionics systems has a very direct impact on flight testing. The older generation of frequency and pulse code modulation instrumentation systems, typically used to acquire avionics data in flight, cannot be used to acquire the large volume of data from these high-speed buses. As a result, each avionics integration contractor develops a unique digital bus instrumentation and data reduction system with little regard for inter-program commonality/compatibility. These systems are not only very expensive, but vary widely with respect to performance characteristics like time correlation accuracy, data compression/alteration algorithms, etc. They may also be one-of-a-kind systems that are integrated with other contractor owned facilities and, if so, cannot be moved to the AFFTC. The AFFTC is working with industry and other government flight test organizations to achieve some standardization in airborne digital bus instrumentation and this effort is a consideration in the IFAST development.

7.2.2 Distributed Processing

Trends toward more distribution of avionics data processing to increasing numbers of embedded processors is reducing the availability of data from the digital integration buses. The end result of this technology impact is that significant amounts of data needed for DT&E will only be accessible from these processors through the ground maintenance connectors or computer monitor and control ports (CMCPs). The CMCPs also have different electrical, mechanical, and data format characteristics from one manufacturer to another and, quite often from one machine to another. Hence, another opportunity for cost effective commonality/compatibility. This goal is also being pursued in the IFAST development and is under consideration by the USAF Deputy for Avionics Control for potential standardization in some manner.

7.2.3 Programmable Software

The capability to change functions/performance of avionics virtually overnight by reprogramming software places a substantial burden on DT&E personnel. Timely assessments of the effects software changes may have on functions that have already been evaluated in-flight are very difficult, if not impossible, to achieve without automated support tools and simulation data which can be used for relative comparisons. Another very significant impact to flight test comes from the increasingly large number of software problems that can be expected to occur during FSD flight testing as a direct result of the correspondingly large increase in the amount of programmable software. Reasonable predictions can be made of the number of software problems that would require extra flights if no IFAST type facility were available and it might be beneficial to make one that shows the significance of this technology impact. For an avionics system with 640K words of programmable software, a trial estimate indicated that major software problems/changes would require 120 flights. This prediction covered the complete flight test period from pre-baseline development through post-baseline evaluation and it only addressed software problems. A conservation cost for these FSD flights exceeds five-million dollars.

7.2.4 New Applications

More advanced weapon systems are coming to the AFFTC that incorporate new avionics technologies like multi-function displays, voice control, integrated fire and flight control, global positioning, sensor data fusion, etc. New innovative DT&E techniques and support resources will be needed to insure the success of these programs. These new applications will impact the methodology employed in flight testing activities like:

(1) Test planning.

(2) Data acquisition.

(3) Problem resolutions.

(4) Data management, analysis and reporting.

They will also have a very significant impact on facilities like the IFAST from the standpoint that moderate IFAST development will be needed continuously to keep the ADPS and avionics interfaces up-to-date.

7.3 Building Description

The IFAST building is a three-story, 52,000 square-foot structure that is located southwest of Base Operations. The second and third floors are each configured to provide two identical, user program areas. The IFAST cadre on the first floor can provide general support to any of the four program areas or any of the four can be utilized in a self-contained fashion. The first floor contains rooms for work breaks, conferences/training, engineering analysis, etc., for all building occupants. Each user area on the upper floors has a shielded bay which can be opened for emitters to scan targets outside the building. The half-sphere domes, shown in Figure 1 at each end of the top floor, were installed for this purpose. The building site and orientation were chosen to provide each bay with an unobstructed view of targets of opportunity in the normal aircraft traffic patterns. This design feature greatly reduces the number of dedicated target missions that would have to be flown otherwise. Other general features of the building include:

(1) Automated security with card-key access control.

(2) Large freight elevator to all floors and roof.

(3) Heavy load bearing roof with anchor fittings for large antennas.

(4) Integrated communications within the building, on base, and with ARPANET.

Other features of each user area are:

(1) Avionics utilities.

(2) Raised technical flooring.

(3) Engineering/administrative space.

(4) Shop and storage areas.

(5) Halon fire protection.

7.4 Automated Data Processing Systems

Development of the IFAST ADPS is being done with a phased approach under direct government management. Lessons learned from each phase are fed back into the requirements of each succeeding phase. The following list of phases provides an overview for development of a time-shared computer complex and the initial test bay computer complex.

(1) Installation and checkout of basic systems.

(2) Rehost of government owned simulation support software.

(3) Development of cockpit operator station (COS).

(4) Real-time simulation demonstration with real avionics.

(5) Initial user program adaptation.

(6) Development/installation of avionics software tools.

(7) Development of generic simulation software.

(8) Development of generic performance monitoring and control systems.

(9) Adaptation to other user programs.

7.4.1 Time-shared Computer Complex

This complex is located on the first floor to support real-time simulation software development, test scenario generation, avionics software development/modifications, data base management, and remote communications. It provides interactive terminals, modems, inter-computer networks, and the following systems:

(1) A Digital Equipment Corporation (DEC) 2060 mainframe, general purpose computer system with two megawords of memory; 2400 megabytes (MB) of disc storage; 800/1600 and 6250/1600 bits-per-inch (bpi) selectable density tape drives; a 900 lines-per-minute (lpm) scientific character printer; a letter quality printer; and remote device interfaces.

(2) A DEC VAX 11/782 simulation development minicomputer system with two central processor units and 3 3/4 MB of memory; 768 MB of disc storage; 6250/1600 bpi selectable density tape drives; a 600 lpm printer/plotter; and a 16-color raster graphics system.

(3) A DEC VAX 11/750 graphics development minicomputer system with 2 MB of memory; 335 MB of disc storage; a 1600 bpi tape unit; hardcopy terminal; Evans and Sutherland (E&S) color picture system; Adage graphics system; and Versatec 22 inch plotter.

7.4.2 Test Bay Computer Complex

Initially only one of these four test bay complexes will be developed by the IFAST program and applied to a DT&E program. The other three bays will be populated by air vehicle prime contractors or avionics subcontractors until more AFFTC expertise is gained and applied to additional DT&E programs with appropriate scheduling. This complex interfaces with the real avionics through the MIL-STD-1553 digital data bus and standard synchro, analog, and discrete signals. The primary systems included in this initial complex include:

(1) Two DEC VAX 11/780 real-time simulation minicomputer systems with 2 MB of memory each and 3 MB of shared memory that can be connected to two more 11/780s; 6520/1600 bpi selectable density tape drives; a 600 lpm printer/plotter; and a 16-color raster graphics system.

(2) A DEC VAX 11/750 minicomputer system for real-time control of an E&S graphics system, with 2 MB of memory; 335 MB of disc storage; E&S color picture system; Adage graphics system; and Versatec 22 inch plotter.

(3) A DEC VAX 11/730 simulation data monitoring and control minicomputer system with 1 MB of memory; 30 MB of disc storage; and avionic systems interfaces.

7.5 Test Bay #1 Initial Implementation

The initial test bay complex implementationn will combine the computer systems described above with an in-house developed generic Cockpit Operator Station (COS) to provide a real-time flight simulation demonstration and avionics test platform. A block diagram of the COS and its connections with the test bay computer system and support equipment is shown in Figure 2.

7.5.1 Cockpit Operator Station

The heart of the IFAST real-time avionics system development is the COS which consists of a cockpit mockup which contains data acquisition and control system and F-16 avionics components that were selected for an initial simulation demonstration. The avionics functions incorporated in the COS include:

(1) F-16A Fire Control Computer (FCC)

(2) Fire Control/Navigation Panel (FC/NP)

(3) Head Up Display (HUD)

(4) Actual and simulated flight instruments

(5) Sidestick controller, throttle, and rudder pedals

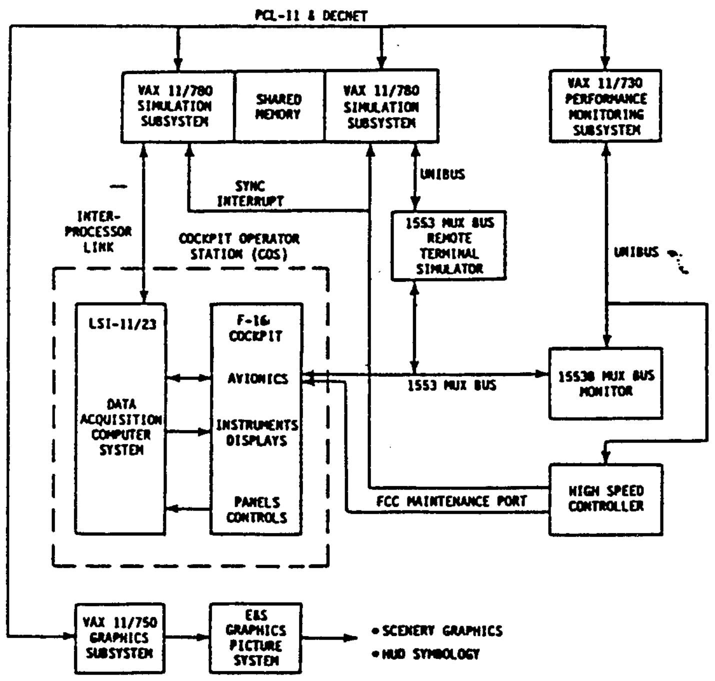

PCL-11 & DECNET
VAX 11/780 SIMULATION SUBSYSTEM
SHARED MEMORY
VAX 11/780 SIMULATION SUBSYSTEM
VAX 11/730 PERFORMANCE MONITORING SUBSYSTEM
UNIBUS
SYNC INTERRUPT
INTER-PROCESSOR LINK
COCKPIT OPERATOR STATION (COS)
1553 MUX BUS REMOTE TERMINAL SIMULATOR
UNIBUS
LSI-11/23
DATA ACQUISITION COMPUTER SYSTEM
F-16 COCKPIT
AVIONICS
INSTRUMENTS DISPLAYS
PANELS CONTROLS
1553 MUX BUS
1553B MUX BUS MONITOR
FCC MAINTENANCE PORT
HIGH SPEED CONTROLLER
VAX 11/750 GRAPHICS SUBSYSTEM
E&S GRAPHICS PICTURE SYSTEM
• SCENERY GRAPHICS
• HUD SYMBOLOGY

Figure 2. Test Bay #1 Initial Configuration

(6) Simulation Control Panel

(7) Simulated Multi-Function Display

Electrical interfaces with the cockpit are handled by the LSI-11/23 based Data Acquisition Computer System (DACS). The DACS incorporates digital, analog, synchro, and discrete signal conditioning for cockpit components. Generic growth capability is designed into the system via a modular chassis concept and plug-in interface circuit cards.

The DACS software controls cockpit interface functions and is adaptable to most hardware configurations. In the initial implementation, the software performs:

(1) System readiness testing

(2) Instrument and control calibrations

(3) Instrument data conversion and scaling

(4) Simulation control and operator input recognition

(5) Data formatting/communication with VAX simulation system

7.5.2 Interprocessor Link

The interprocessor link provides a high speed parallel data path between the simulation system and the DACS. Utilizing a common memory interface, this link supports software download, bidirectional simulation data transfer, and sync interrupts from the VAX 11/780 to the LSI-11/23.

7.5.3 Simulation Subsystem

The dual VAX 11/780 computers host the aircraft aerodynamic and basic avionic system simulation models. The avionics model packages include the Inertial Navigation System, Instrument Landing System, Stores Management System, HUD, and weapons operation and scoring. A programmable multiplex bus terminal connects the simualtion computers to the 1553B Mux Bus employed by the FCC and FC/NP. By responding on the bus in real time with simulated avionics data, the simulation subsytem can provide realistic flight performance dynamics which are repeatable and easily instrumented.

7.5.4 Graphics Subsystem

The graphics computer system receives simulated aircraft and target flight data and HUD display information formthe simulation subsystem which it uses to drive a display situated in front of the cockpit. This display provides out-the-window scenery graphics with a superimposed HUD display (if selected) for flight orientation and landmark recognition.

7.5.5 Performance Monitoring Subsystem

In addition to the VAX 11/730 computer, the performance monitoring subsystem (PMS) includes a programmable multiplex bus terminal configured to monitor data activity on the 15538 Mux Bus. Bus traffic may be selectively displayed and recorded for later reduction and analysis.

The PMS also includes a high speed controller (HSC) that is connected to the maintenance port of the FCC. This device permits control and breakpoint monitoring of the FCC during real time simulation to generate sync interrupts to the simulation subsystem software. The HSC is also used to download an operational flight program (OFP) into the FCC.

7.6 Utilization

The primary purpose of the IFAST is to support avionics flight testing. As such, the first order of business is assessment of avionics flight readiness. If problems occur before or during a flight, the second order of business is a quick assessment to determine what subsequent actions are appropriate. This test-before-fly process uses real-time simulation in IFAST to provide an integrated checkout of the avionics. In order to insure the validity of assessments, the avionics configurations and simulations must be reasonably accurate, and if so, dynamic evaluation of the avionics in simulated flight conditions can be done through the use of sophisticated performance monitoring and control in IFAST.

The capability for testing the avionics prior to flight results in several high payoff areas. A few of the more important areas are:

(1) Safety of flight validation

(2) Crew familiarization and smart test planning

(3) Acquisition of test data for simulation baseline performance

(4) Reduction of system integration time.

The role of the IFAST in these important areas is to provide a major benefit to test programs by allowing the expensive flight test hours to be used for gathering meaningful data to evaluate weapon systems performance. Through the use of the real-time simulation capability, users can evaluate the performance of each subsystem in a system context and can better understand the complex interactions of the subsystems.

The value of using a flight test support facility like the IFAST must be measured in consideration of the efficiency and completeness it offers as a supplement to flight testing, as well as the relatively minimal cost of operating it. However, the high payoff comes from the increases in quality of the products placed in operational service.

7.7 References

1. "New Analytical Tools," Aviation Week & Space Technology, January 17, 1983.