MIL-STD-1553

Protocol Tutorial

MIL-STD-1553 Protocol Tutorial

Document Date: 12 May, 2005
Document Revision: 1.0

ZhengHong Aviation Tech Co.,ltd
#1 TaiBai North Rd. Xi'an 710068 China

86-029-84288198, 84288197
86-029-84288198-8815 (fax)
info@zhhktech.com (email)
http://www.zhhktech.com
http://www.1553b-arinc429.com

Contents and Tables

Contents

Chapter 1 Overview

Interface Description 1
Background 2
The Advent of Digital Technology 2
The Advent of the Data Bus 3
History and Applications 5
Notice 1 and Notice 2 5
MIL-STD-1553 Applications 6
MIL-STD-1553B Defined 6

Chapter 2 Hardware Elements

Transmission Media 8
Multi-stub Couplers 11
Remote Terminals 11
Bus Controller 13
Word Controller 14
Message Controller 14
Frame Controller 14
Bus Monitor 15
Terminal Hardware 16

Chapter 3 Protocol

Word Types 17
Sync Fields 19
Command Words 19
Data Word 21
Status Word 21
Resetting of the Status Word 21
Terminal Address 22
Message Error 22
Instrumentation 22

Service Request 23
Reserved 23
Broadcast Command Received 23
Busy 24
Subsystem Flag 24
Dynamic Bus Control Acceptance Bit 25
Terminal Flag 25

Chapter 4 Message Formats

Introduction 26
Bus Controller to Remote Terminal 28
Remote Terminal to Bus Controller 28
Remote Terminal to Remote Terminal 28
RT-RT Validation 29
Mode Command Formats 29
Broadcast Information Transfer Formats 30
Command and Message Validation 30
Illegal Commands 31
Example 1 31
Example 2 32
Example 3 32
Terminal Response Time 32
Inter-message Gap 33
Superseding Commands 33

Chapter 5 Mode Codes

Introduction 35
Mode Code Identifier 36
Dynamic Bus Control 37
Synchronize 38
Transmit Status Word 38
Initiate Self-test 39
Transmitter Shutdown 40
Override Transmitter Shutdown 40
Inhibit Terminal Flag 40
Override Inhibit Terminal Flag 41
Reset Remote Terminal 41
Transmit Vector Word 42
Synchronize with Data Word 42
Transmit Last Command Word 43
Transmit Built-in-Test (BIT) Word 43
Selected Transmitter Shutdown 44
Override Selected Transmitter Shutdown 44

Reserved Mode Codes....44
Required Mode Codes....45
Broadcast Mode Codes....46

Chapter 6 System Issues

Using Subaddresses....47
Extended Subaddressing....47
Data Wrap-Around....48
Data Buffering....49
Variable Message Blocks....50
Sample Consistency....50
Data Validation....50
Major/Minor Frame Timing....51
Bus Loading....53
Error Processing....54

Chapter 7 Connecting the Bus

Introduction....56
Direct Coupled Bus....56
Transformer Coupled Bus....57
Mixed Bus Coupling....58
Primary and Secondary Buses....58
More than Two Terminals....59
Cautions....60
The Single Bus Wiring Short Cut....60
The Crossed Bus Wiring Short Cut....61
The Two Bus/One Controller Short Cut....61
The Very Long Stub Short Cut....61
The No Terminators Short Cut....61
The Connected Couplers Short Cut....62
The Junk Wiring Short Cut....62

Chapter 8 Testing

Introduction....63
Government and SAE Test Plans....63
Test Equipment....64

Chapter 9 Additional Information

Sources of Information....65

Table of Figures

Figure 1. System Configurations 4
Figure 2. Terminal Connection Methods 10
Figure 3. Simple Multiplex Architecture 12
Figure 4. Terminal Definition 13
Figure 5. Word Formats 18
Figure 6. Data Encoding and Decoding 19
Figure 7. Information Transfer Formats 27
Figure 8. Information Transfer Formats (Broadcast) 30
Figure 9. Major/Minor Cycles 53
Figure 10. Direct Coupling 57
Figure 11. Transformer Coupling 57
Figure 12. Mixed Bus Coupling 58
Figure 13. Primary and Secondary Buses 59

List of Tables

Table 1. Summary of MIL-STD-1553 Characteristics 7
Table 2. Summary of Transmission Media Characteristics 9
Table 3. Terminal Electrical Characteristics Input Characteristics 16
Table 4. Terminal Electrical Characteristics Input Characteristics 16
Table 5. Mode Codes 36
Table 6. Bus Loading Numbers 53
Table 7. Test Plans 63

CHAPTER 1

1. Overview

Interface Description

MIL-STD-1553 is a military standard that defines the electrical and protocol characteristics for a data bus. A data bus is used to provide a medium for the exchange of data and information between various systems. It is similar to what the personal computer and office automation industry has dubbed a Local Area Network (LAN).

This guide provides an introduction to the MIL-STD-1553 data bus, its history, applications, and use. It describes:

- The physical elements that make up the bus.
- The protocol, including the message formats, word types, and command and status words.
- Status word bits and mode commands and their definitions and use, both from the remote terminal and bus controller perspective.
- Issues such as bus loading, major and minor frame timing, and error recovery.

Additionally, this guide introduces the various test plans and discusses how the Condor MIL-STD-1553 product line can solve some of your testing and application needs.

Background

In the 1950s and 1960s, aviation electronics, referred to as avionics, were simple stand-alone systems. The navigation, communications, flight controls, and displays consisted of analog systems. Often these systems were composed of multiple boxes, or subsystems, connected to form a single system. Various boxes within a system were connected with point-to-point wiring. The signals mainly consisted of analog voltages, synchro-resolver signals, and switch contacts. The location of these boxes within the aircraft was a function of operator need, available space, and the aircraft weight and balance constraints. As more and more systems were added, the cockpits became more crowded, the wiring more complex, and the overall weight of the aircraft increased.

By the late 1960s and early 1970s, it became necessary to share information between the various systems to reduce the number of black boxes required by each system. A single sensor, for example that provided heading and rate information, could provide that data to the navigation system, the weapons system, the flight control system, and pilots display system (see figure 1a).

However, the avionics technology was still basically analog, and while sharing sensors did produce a reduction in the overall number of black boxes, the connecting signals became a “rat's nest” of wires and connectors. Moreover, functions or systems that were added later became an integration nightmare, as additional connections of a particular signal could have potential system impacts. Additionally, as the system used point-to-point wiring, the system that was the source of the signal typically had to be modified to provide the additional hardware to output to the newly added subsystem. As such, inter-system connections had to be kept to the bare minimum.

The Advent of Digital Technology

By the late 1970s, with the advent of digital technology, digital computers had made their way into avionics systems and subsystems. They offered increased computational capability and easy growth, compared to their analog predecessors. However, the data signals, inputs and outputs from the sending and receiving systems were still mainly analog in nature. This led to the configuration of a small number of centralized computers (typically only one or two) being interfaced to other systems and subsystems via complex and expensive analog-to-digital and digital-to-analog converters.

As time and technology progressed, the avionics systems became more digital. And with the advent of the microprocessor, things really took off. A benefit of this digital application was the reduction in the number of analog signals, and hence the need for their conversion. Transferring the

data between users in digital form could provide a greater sharing of sensor information. An additional side benefit was that digital data could be transferred bi-directionally, wherein analog data was transferred unidirectional. Serial rather than parallel transmission of the data was used to reduce the number of interconnections within the aircraft and the receiver/driver circuitry required with the black boxes.

The Advent of the Data Bus

But this alone was still not enough. A data transmission medium, which would allow all systems and subsystems to share a single and common set of wires, was needed (see figure 1b). By sharing the use of this interconnect, the various subsystems could send data between themselves and to other systems and subsystems, one at a time, and in a defined sequence, hence a data bus.

MIL-STD-1553B defines the term Time Division Multiplexing (TDM) as "the transmission of information from several signal sources through one communications system with different signal samples staggered in time to form a composite pulse train." For our example in Figure 1b, this means that data can be transferred between multiple avionics units over a single transmission media, with the communications between the different avionics boxes taking place at different moments in time, hence time division.

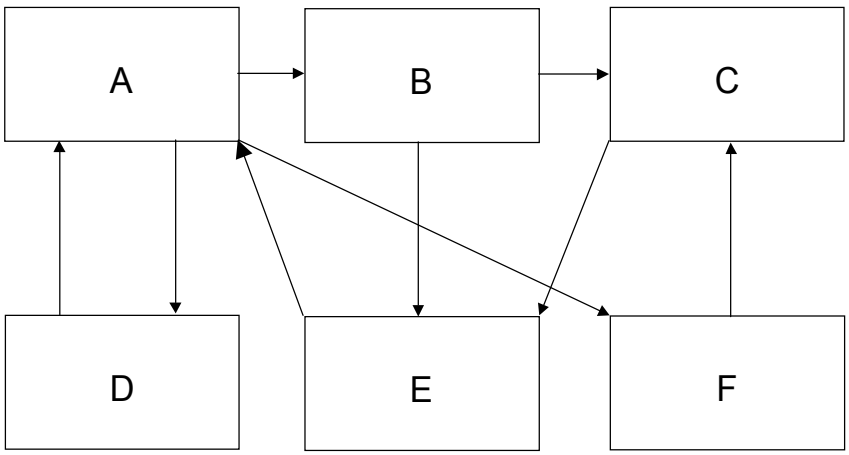

Figure 1A

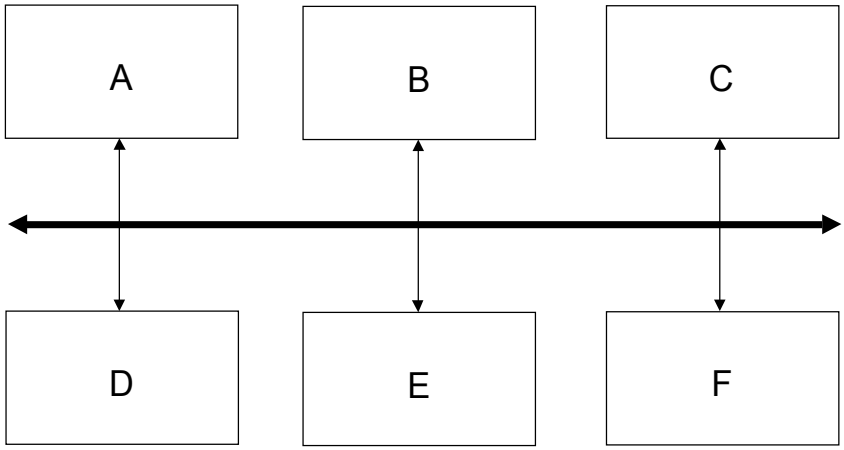

Figure 1B

Figure 1. System Configurations

History and Applications

In 1968 the Society of Automotive Engineers (SAE), a technical body of military and industrial members, established a subcommittee to define a serial data bus to meet the needs of the military avionics community. Known as the A2-K, this subcommittee developed the first draft of the document in 1970. Three years of military and government reviews and changes led to the release of MIL-STD-1553 (USAF) in August of 1973. The primary user of the initial standard was the F-16.

Further changes and improvements were made and a tri-service version, MIL-STD-1553A was released in 1975. The first users of the "A" version of the standard was the Air Force's F-16 and the Army's new attack helicopter, the AH-64A Apache.

With some "real world" experience, it was soon realized that further definitions and additional capabilities were needed. The SAE spent three years of concentrated effort to produce 1553B, which was released in 1978. At that point, the government decided to "freeze" the standard at the "B" level so as to allow component manufacturers to develop products and to allow industry to gain some additional "real world" experience before determining what the next set of changes were to be made.

Notice 1 and Notice 2

Today, the 1553 standard are still at the "B" level. However, changes have been made. In 1980, the Air Force introduced Notice 1. Intended only for Air Force applications, Notice 1 restricted the use of many of the options within the standard. While the Air Force felt this was needed to obtain a common set of avionics systems, many in industry felt that Notice 1 was too restrictive and limited the capabilities in the application of the standard.

The SAE again spent over three years in committee work to develop Notice 2. Released in 1986, the tri-service Notice 2 (which supersedes Notice 1) places tighter definitions upon the options within the standard. And while not restricting an option's use, it tightly defines how an option is to be used if implemented. Notice 2—in an effort to obtain a common set of operational characteristics—also places a minimum set of requirements upon the design of the black box.

MIL-STD-1553 Applications

Since its inception, MIL-STD-1553 has found numerous applications. Notice 2 to the standard has even removed all references to “aircraft” or “airborne” so as not to limit its use. While all the programs to which the standard has been applied are too numerous to be covered here, the following is a summary of its uses.

While the standard has been applied to satellites as well as payloads within the space shuttle (it is even being used on the International Space Station), its military applications are the most numerous and far ranging. It has been employed on large transports, aerial refuelers, and bombers, tactical fighters, and helicopters. It is even contained within missiles and serves, in some instances, as the primary interface between the aircraft and a missile. The Navy has applied the data bus to both surface and subsurface ships. The Army, in addition to its helicopters, has put 1553 into tanks and howitzers.

Commercial applications have applied the standard to systems including subways, for example the Bay Area Rapid Transit (BART), and manufacturing production lines.

MIL-STD-1553B has also been accepted and implemented by NATO and many foreign governments. The UK has issued Def Stan 00-18 (Part 2) and NATO has published STANAG 3838 AVS, both of which are versions of MIL-STD-1553B.

The broad acceptance and application of MIL-STD-1553B has also fostered the development of other standardization efforts. MIL-STD-1773 is a fiber optic version of 1553B. And MIL-STD-1760A, the Aircraft/Store Interconnect, has 1553B embedded within it.

MIL-STD-1553B Defined

So now that we understand the driving need for the development of a data bus, and a little of its history and application, what exactly is MIL-STD-1553B? A summary of the characteristics of MIL-STD-1553B is found in Table 1.

Table 1. Summary of MIL-STD-1553 Characteristics

Data Rate	1 MHz
Word Length	20 bits
Data Bits / Word	16 bits
Message Length	Maximum of 32 data words
Transmission Technique	Half-duplex
Operation	Asynchronous
Encoding	Manchester II bi-phase
Protocol	Command/response
Bus Control	Single or Multiple
Fault Tolerance	Typically Dual Redundant, second bus in “Hot Backup” status
Message Formats	Controller to terminal Terminal to controller Terminal to terminal Broadcast System control
Number of Remote Terminals	Maximum of 31
Terminal Types	Remote terminal Bus controller Bus monitor
Transmission Media	Twisted shielded pair
Coupling	Transformer and direct

The primary purpose of the data bus is to move data between black boxes. How these boxes are connected and the methodology with which the communication is accomplished is central to the operation of the data bus.

However, before we delve into to the protocol, it is necessary to understand a little of the data bus hardware.

2. Hardware Elements

MIL-STD-1553 defines certain aspects regarding the design of the data bus system and the black boxes to which the data bus is connected. The standard defines four hardware elements. These are:

- The transmission media.
- Remote terminals.
- Bus controllers.
- Bus monitors.

Transmission Media

The transmission media, or data bus, is defined as a twisted shielded pair transmission line consisting of the main bus and a number of stubs. There is one stub for each terminal connected to the bus. The main data bus is terminated at each end with a resistance equal to the cable's characteristic impedance (plus or minus two percent). This termination makes the data bus behave electrically like an infinite transmission line.

Stubs, which are added to the main bus to connect the terminals, provide "local" loads and produce impedance mismatch where added. This mismatch, if not properly controlled, produces electrical reflections and degrades the performance of the main bus. Therefore, the characteristics of both the main bus and the stubs are specified within the standard. Table 2 summarizes the transmission media characteristics.

Table 2. Summary of Transmission Media Characteristics

Cable Type	Twisted Shielded Pair
Capacitance	30.0 pF/ft. max, wire to wire
Characteristic Impedance	70.0 to 85.0 Ohms at 1 MHz
Cable Attenuation	1.5 dB/100 ft. max, at 1MHz
Cable Twists	4 Twists per ft., minimum
Shield Coverage	90% min (Notice 2)
Cable Termination	Cable impedance (+/- 2%)
Direct Coupled Stub Length	Maximum of 1 foot
XFMR Coupled Stub Length	Maximum of 20 feet

There is now no maximum length for the main data bus. Following transmission line design practices and careful placement of the stubs can yield working systems with the main bus length of several hundred meters. It is highly recommended that the bus be modeled and tested to insure its operation and performance characteristics.

The standard specifies two methods for a terminal to be connected to the main bus:

- Direct coupled
- Transformer coupled

Figure 2 shows the two methods. The primary difference between the two being that the transformer coupled method uses an isolation transformer for connecting the stub cable to the main bus cable. In both methods, two isolation resistors are placed in series with the bus. In the direct-coupled method, the resistors are typically located within the terminal, whereas in the transformer-coupled method, the resistors are typically located with the coupling transformer in boxes called data bus couplers.

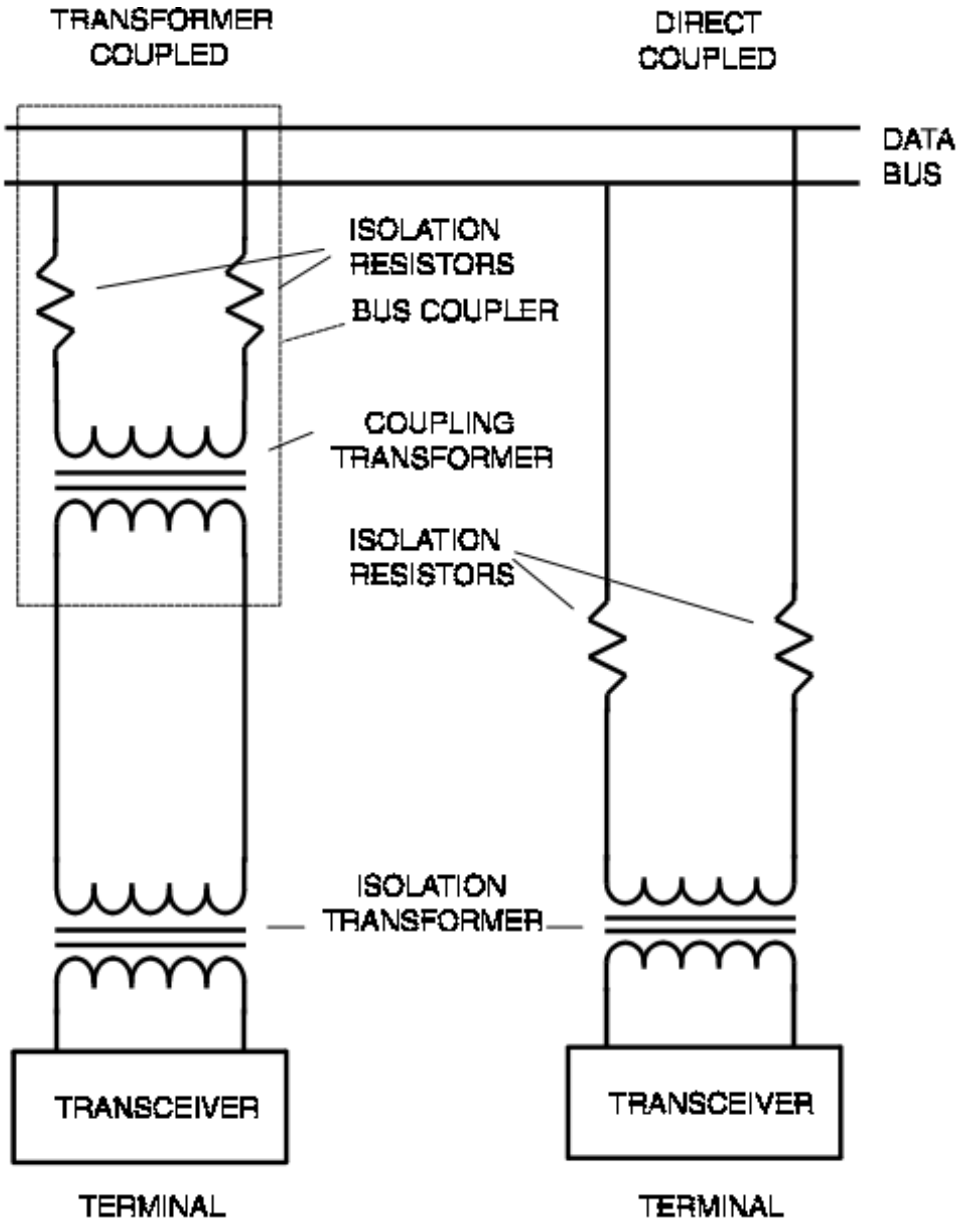

Figure 2. Terminal Connection Methods

Another difference between the two coupling methods is the length of the stub. For the direct-coupled method, the stub length is limited to a maximum of one foot. For the transformer-coupled method, the stub can be up to a maximum of twenty feet long. Therefore for direct-coupled systems, the data bus must be routed in close proximity to each of the terminals, whereas for a transformer-coupled system, the data bus may be up to twenty feet away from each terminal.

Buses using direct-coupled stubs must be designed to withstand the impedance mismatches caused by the placement of the stubs. Thus you should perform a careful tradeoff analysis to determine where stubs are

placed. You should build and test designs using the direct-coupled method to determine if the waveform distortion causes receiver problems.

Multi-stub Couplers

In the past, it was normal to have one data bus coupler (transformer coupling) on the data bus for each black box connected. In recent years, multi-stub couplers have been introduced and are gaining acceptance. The use of a multi-stub coupler reduces the number of separate components that must be procured and reduces the aircraft wiring. In some applications today, one multi-stub coupler is provided for each equipment bay or rack. An additional "spare" stub can be supplied at each coupler for connection of monitors and test equipment for troubleshooting and maintenance.

Remote Terminals

Remote terminals are defined within the standard as "All terminals not operating as the bus controller or as a bus monitor".Therefore if it is not a controller, monitor, or the main bus or stub, it must be a remote terminal. The remote terminal comprises the electronics necessary to transfer data between the data bus and the subsystem. So what is a subsystem? For 1553 applications, the subsystem is the sender or user of the data being transferred.

In the earlier days of 1553, remote terminals were used mainly to convert analog and discrete data to and from a data format compatible with the data bus. The subsystems were still the sensor that provided the data and computer, which used the data. As more and more digital avionics became available, the trend has been to embed the remote terminal into the sensor and computer. Today it is common for the subsystem to contain an embedded remote terminal. Figure 3 shows the different levels of remote terminals.

A remote terminal typically consists of a transceiver, an encoder/decoder, a protocol controller, a buffer or memory, and a subsystem interface. In a modern black box containing a computer or processor, the subsystem interface may consist of the buffers and logic necessary to interface to the computer's address, data, and control buses. For dual redundant systems, the most prevalent in today's applications, two transceivers and two encoders/decoders would be required. Figure 4 is a block diagram of a remote terminal and its connection to a subsystem. It should be pointed out that if the remote terminal shares common memory (verses private), then that portion of the memory that the remote terminal can address should be considered part of the remote terminal (as indicated in Figure 4). The remote terminal consists of all the electronics necessary to transfer data

between the data bus and the user or originator of the data being transferred.

But a remote terminal must be more than just a data formatter. It must be capable of receiving and decoding commands from the bus controller and responding accordingly. It must also be capable of buffering a message worth of data, detecting transmission errors and performing validation tests upon the data, and reporting the status of the message transfer. Notice 2 to the standard also requires that a remote terminal be capable of performing a few of the bus management commands (referred to as mode commands). For dual redundant applications, it must be capable of listening to, and decoding, commands on both buses at the same time.

A remote terminal *must* follow the protocol defined by the standard. It can only respond to commands received from the bus controller (i.e., it speaks only when spoken to). When it receives a valid command, it must respond within a very small, closely defined amount of time. If a message doesn't meet the validity requirements defined, then the remote terminal must invalidate the message and discard the data (not allow it to be used by the subsystem). In addition to reporting status to the bus controller, most remote terminals today are capable of providing some level of status information to the subsystem.

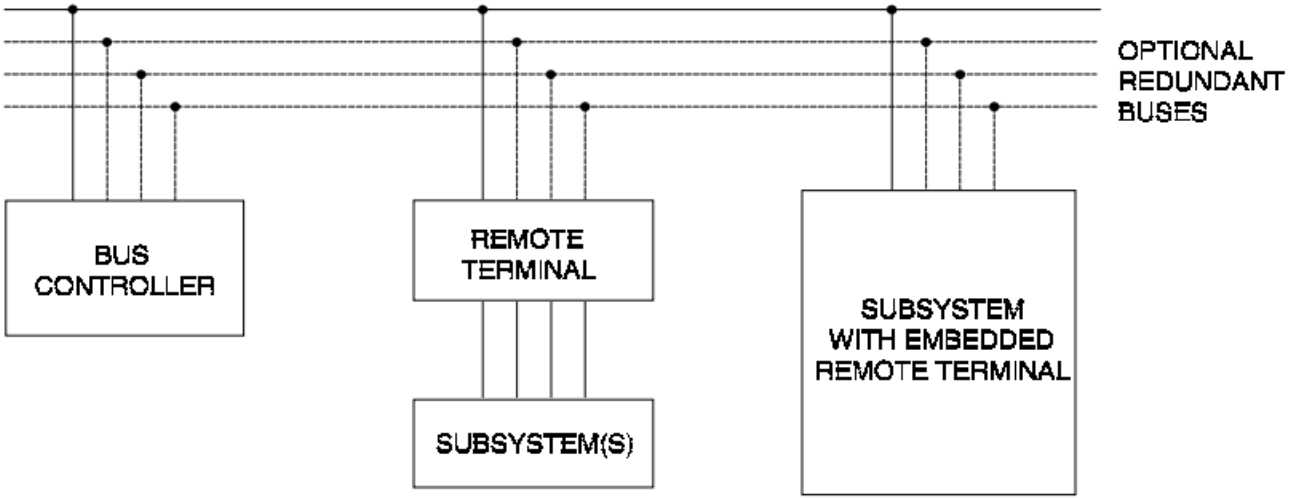

Figure 3. Simple Multiplex Architecture

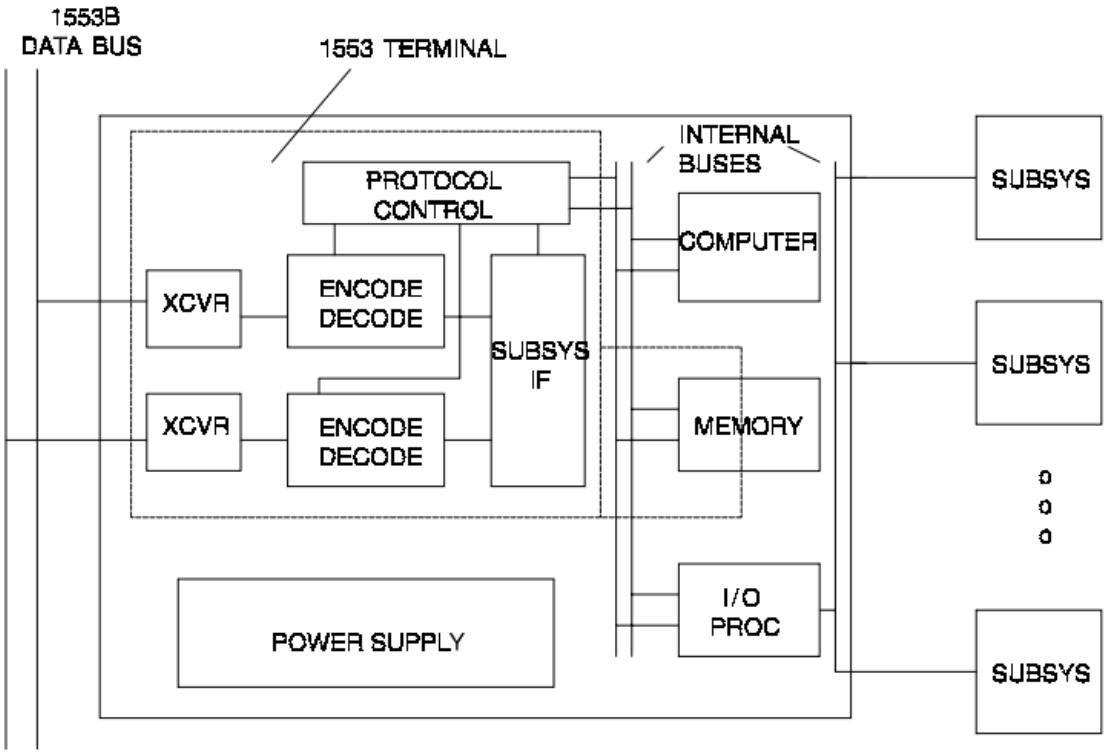

Figure 4. Terminal Definition

Bus Controller

The bus controller is responsible for directing the flow of data on the data bus. While several terminals may be capable of performing as the bus controller, only one bus controller may be active at a time. The bus controller is the only one allowed to issue commands onto the data bus. The commands may be for the transfer of data or the control and management of the bus (referred to as mode commands).

Typically, the bus controller is a function that is contained within some other computer, such as a mission computer, a display processor, or a fire control computer. The complexity of the electronics associated with the bus controller is a function of the subsystem interface (the interface to the computer), the amount of error management and processing to be performed, and the architecture of the bus controller.

There are three types of bus controller architectures:

- A word controller.
- A message controller.

- A frame controller.

There is no requirement within the standard as to the internal workings of a bus controller, only that it issue commands on to the bus.

Word Controller

A word controller is the oldest and simplest type of controller. Few word controllers are built today. For a word controller, the terminal electronics transfers one word at a time to the subsystem. Message buffering and validation must be performed by the subsystem. This places quite a burden on the subsystem due to the timing requirements of the bus.

Message Controller

Today, many of the "fielded" bus controllers are message controllers. These controllers output a single message at a time, interfacing with the computer only at the end of the message or perhaps when an error occurs. Some message controllers are capable of performing minor error processing, such as transmitting once on the alternate data bus, before interrupting the computer.

The computer informs the interface electronics where the message exists in memory and provides a control word. For each message, the control word typically informs the electronics of the message type (e.g., a RT-BC or RT-RT command), on which bus to transfer the message, where to read or write the data words in memory, and what to do if an error occurs. The control words are a function of the hardware design of the electronics and aren't standardized among bus controllers.

Frame Controller

A frame controller is the latest concept in bus controllers. With the advent of microprocessors and Application Specific Integrated Circuits (ASIC's), this type of controller is rapidly becoming the norm. A frame controller is capable of processing multiple messages in a sequence defined by the host computer. The frame controller is typically capable of performing some error processing as defined by the message control word. Frame processors are used to "off load" the subsystem or host computer as much as possible, interrupting only at the end of a series of messages or when an error that it cannot handle is detected.

Bus Monitor

A bus monitor is a terminal that listens (monitors) to the exchange of information on the data bus. The standard strictly defines how bus monitors may be used, stating that the information obtained by a bus monitor be 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." A monitor may collect all the data from the bus or may collect selected data. The reason for restricting its use is that while a monitor may collect data, it deviates from the command-response protocol of the standard, in that a monitor is a passive device that doesn't transmit a status word and therefore cannot report on the status of the information transferred. Bus monitors fall into two categories:

- A recorder for testing.
- A terminal functioning as a back-up bus controller.

In collecting data, a monitor must perform the same message validation functions as the remote terminal and if an error is detected, inform the subsystem of the error (the subsystem may still record the data, but the error should be noted). For monitors, which function as recorders for testing, the subsystem is typically a recording device such as a magnetic tape or disk, or a telemetry transmitter. For monitors, which function as back-up bus controllers, the subsystem is the computer.

Today it is common for bus monitors to contain a remote terminal. When the monitor receives a command addressed to its terminal address, it responds as a remote terminal. For all other commands, it functions as a monitor. The remote terminal portion can be used to provide feedback to the bus controller, which monitors status and the amount of memory or time (i.e. recording tape) left. The remote terminal portion also can be used to reprogram a selective monitor as to what messages to capture.

Terminal Hardware

The electronic hardware between a remote terminal, bus controller, and bus monitor doesn't differ much. Both the remote terminal and bus controller (and bus monitor if it is also a remote terminal) must have the transmitters/receivers and encoders/decoders to format and transfer data. The requirements upon the transceivers and the encoders/decoders don't vary between the hardware elements. Table 3 and Table 4 list the electrical characteristics of the terminals.

All three elements have some level of subsystem interface and data buffering. The primary difference lays in the protocol control logic and often this just a different series of microcoded instructions. For this

reason, it is common to find 1553 hardware circuitry that is also capable of functioning as all three devices.

There is an abundance of "off-the-shelf" components available today from which to design a terminal. These vary from discrete transceivers, encoders/decoders, and protocol logic devices to a single dual redundant hybrid containing everything but the transformers.

Table 3. Terminal Electrical Characteristics Input Characteristics

Requirements	Transformer Coupled	Direct Coupled	Condition
Input Level	0.86-14.0 V	1.2-20.0 V	p-p, l-l
No Response	0.0-0.2 V	0.0-0.28 V	p-p, l-l
Zero Crossing Stability	+-150.0 nSec	+-150.0 nSec	
Rise/Fall Times	0 nSec - Sine	0 nSec - Sine	
Noise Rejection	140.0 mV WGN	200.0 mV WGN	BER 1 per 10^7
Common Mode Rejection	+- 10.0 V peak	+- 10.0 V peak	line-ground dc-2MHz
Input Impedance	1000 ohms	2000 ohms	75 kHz - 1 MHz

Table 4. Terminal Electrical Characteristics Input Characteristics

Requirements	Transformer Coupled	Direct Coupled	Condition
Output Level	18.0-27.0 V	6.0-9.0 V	p-p, l-l
Zero Crossing Stability	25.0 nSec	25.0 nSec	
Rise/Fall Times	100-300 nSec	100-300 nSec	10%-90%
Max Distortion	+-900.0 mV	+-300.0 mV	peak, l-l
Max Output Noise	14.0 mV	5.0 mV	rms, l-l
Max Residual Voltage	+-250.0 mV	+-90.0 mV	peak, l-l

CHAPTER 3

3. Protocol

Now that you understand a little of the hardware requirements, it's time to discuss the methodology by which the information transfer occurs. The rules under which the transfers occur are referred to as "protocol".

The control, data flow, status reporting, and management of the bus are provided by three word types.

Word Types

Three distinct word types are defined by the standard. These are:

- Command words.
- Data words.
- Status words.

Each word type has a unique format, yet all three maintain a common structure. Each word is twenty bits in length. The first three bits are used as a synchronization field, thereby allowing the decode clock to re-sync at the beginning of each new word. The next sixteen bits are the information field and are different between the three word types. The last bit is the parity bit. Parity is based on odd parity for the single word. The three word types are shown in Figure 5.

Bit encoding for all words is based on bi-phase Manchester II format. The Manchester II format provides a self-clocking waveform in which the bit sequence is independent. The positive and negative voltage levels of the Manchester waveform is DC-balanced (same amount of positive signal as there is negative signal) and, as such, is well suited for transformer coupling.

The Manchester waveform is shown in Figure 6. A transition of the signal occurs at the center of the bit time. A logic "0" is a signal that transitions from a negative level to a positive level. A logic "1" is a signal that transitions from a positive level to a negative level.

It is important to note that the voltage levels on the bus are *not* the signaling media, and that it is strictly the timing and polarity of the zero-crossings that convey information on the bus. For this reason the 1553 bus is extremely forgiving of conditions that cause the voltage levels on the bus to vary.

The terminal's hardware is responsible for the Manchester encoding and decoding of the word types. The interface that the subsystem sees is the 16-bit information field of all words. The sync and parity fields aren't provided directly. However, for received messages, the decoder hardware provides a signal to the protocol logic as to the sync type the word was and as to whether parity was valid or not. For transmitted messages, there is input to the encoder defining what sync type to place at the beginning of the word. The encoder automatically calculates parity.

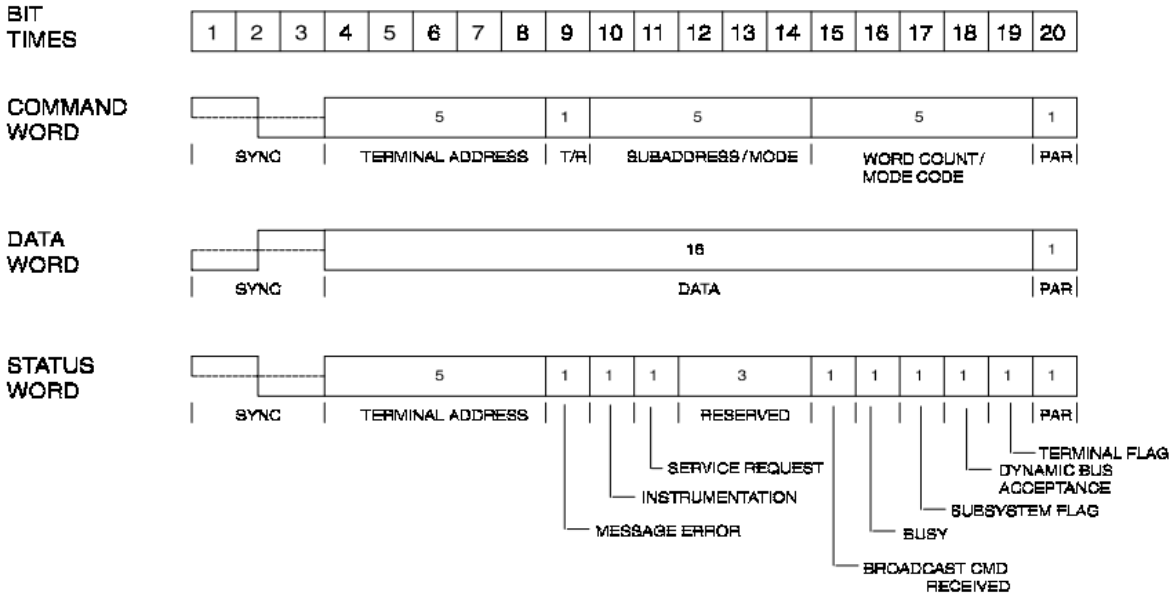

Figure 5. Word Formats

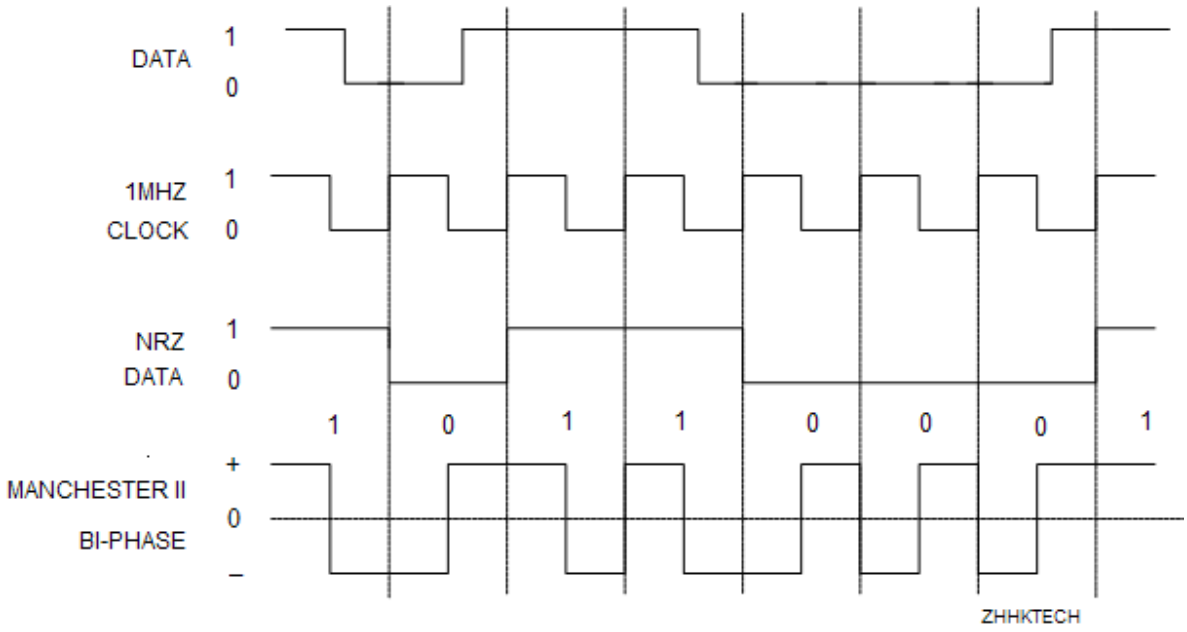

Figure 6. Data Encoding and Decoding

Sync Fields

The first three bit times of all word types is called the sync field. The sync waveform is in itself an invalid Manchester waveform as the transition only occurs at the middle of the second bit time. The use of this distinct pattern allows the decoder to re-sync at the beginning of each word received and maintains the overall stability of the transmissions.

Two distinct sync patterns are used: the command/status sync, and the data sync. The command/status sync has a positive voltage level for the first one and a half bit times, then transitions to a negative voltage level for the second one and a half bit times. The data sync is the opposite, a negative voltage level for the first one and a half bit times, and then a positive voltage level for the second one and a half bit times. The sync patterns are shown in Figure 5.

Command Words

The Command Word (CW) specifies the function that a remote terminal is to perform. Only the active bus controller transmits this word. The word begins with a command sync in the first three bit times. The following sixteen-bit information field is defined in Figure 5. The five bit Terminal Address (TA) field (bit times 4-8) states to which unique remote terminal the command is intended (no two terminals may have the same address).

Note: An address of 00000B is a valid address, and an address of 11111B is *always* reserved as a broadcast address. Additionally, there is no requirement that the bus controller be assigned an address, therefore the maximum number of terminals the data bus can support is 31.

Notice 2 to the standard requires that the terminal address be wire-programmable from an external source to the black box (i.e., an external I/O connector), and that the remote terminal electronics perform a parity test upon the wired terminal address. The Notice basically states that an open circuit on an address line is detected as a logic "1", that connecting an address line to ground is detected as a logic "0", and that odd parity is used in testing the parity of the wired address field.

Why was the parity test added in Notice 2? Some "real world" applications have found that the vibration levels associated with certain functions (e.g., the firing of a missile) can cause intermittent problems with connectors or cause bonding wires within hybrids to open. For this reason most "real world" remote terminals only sample the terminal address wires during power-on or initialization, and don't initialize the terminal if the parity is incorrect.

The next bit (bit time 9) makes up the Transmit/Receive (T/R) bit. This defines the direction of information flow and is *always* from the point of view of the remote terminal. A transmit command (logic 1) indicates that the remote terminal is to transmit data, while a receive command (logic 0) indicates that the remote terminal is going to receive data. The only exceptions to this rule are associated with mode commands.

The next five bits (bit times 10-14) make up the Subaddress (SA)/Mode Command bits. Logic 00000B or 11111B within this field is decoded to indicate that the command is a Mode Code Command. All other logic combinations of this field are used to direct the data to different functions within the subsystem. An example might be that 00001B is position and rate data, 00010B is frequency data, 10010B is display information, and 10011B is self-test data. Use of the subaddresses is left up to you; however, Notice 2 suggests the use of Subaddress 30 for data wrap-around (defined later).

Note: If you are designing a terminal as part of a store, then MIL-STD-1760A places restrictions on the use of certain subaddresses).

The next five bit positions (bit times 15-19) define the Word Count (WC) or Mode Code to be performed. If the Subaddress/Mode Code field is 00000B or 11111B, then this field defines the mode code to be performed. If not a mode code, then this field defines the number of data words to be received or transmitted depending on the T/R bit. A word count field of 00000B is decoded as 32 data words.

The last bit (bit time 20) is the word parity bit. Only odd parity is used.

Data Word

The Data Word (DW) contains the actual information that is being transferred within a message. The first three-bit time contains a data sync. This sync pattern is the opposite of that used for command and status words and therefore is unique to the word type.

Data words can be transmitted by either a remote terminal (transit command) or a bus controller (receive command). Transmit and Receive, by convention, references the remote terminal.

The next sixteen bits of information are left to the designer to define. The only standard requirement is that the most significant bit (MSB) of the data be transmitted first. While the standard provides no guidance as to their use, Section 80 of MIL-HDBK-1553A, Data Word Formats, provides guidance and lists the formats (i.e. bit patterns, resolutions, etc.) of the most commonly used data words. Procurement specifications typically state that Section 80 guidelines should be used in the data word definitions.

The last bit (bit time 20) is the word parity bit. Only odd parity is used.

Status Word

A remote terminal in response to a valid message transmits only the status word (SW). The status word is used to convey to the bus controller whether a message was properly received or to convey the state of the remote terminal (i.e., service request, busy, etc.). The status word is defined in Figure 5.

Since the status word conveys information to the bus controller, there are two views as to the meaning of each bit:

- What the setting of the bit means to a remote terminal
- What the setting of the bit means to a bus controller.

Each field of the status word, and its potential meanings, is examined below.

Resetting of the Status Word

The status word, with the exception of the remote terminal address, is cleared after receipt of a valid command word. The two exceptions to this rule are if the command word received is a Transmit Status Word Mode Code or a Transmit Last Command Word Mode Code. Conditions that set the individual bits of the word may occur at any time. If after clearing the

status word, the conditions for setting the bits still exists, then the bits is set again.

When an error in the data being received is detected, the Message Error bit is set and transmission of the status word is suppressed. Transmission of the status word is also suppressed upon receipt of a broadcast message. For an Illegal Message (i.e. Illegal Command), the Message Error bit is set and the status word is transmitted.

Terminal Address

The first five bits (bit times 4-8) of the information field are the Terminal Address (TA). These five bits should match the corresponding field within the command word that the terminal received. The remote terminal sets these bits to the address to which it has been programmed. The bus controller should examine these bits to insure that the terminal responding with its status word was indeed the terminal to which the command word was addressed.

With a remote terminal to remote terminal message (RT-RT), the receiving terminal should compare the address of the second command word with that of the received status word. While not required by the standard, it is good design practice to insure that the data received is from a valid source.

Message Error

The next bit (bit time 9) is the Message Error (ME) bit. This bit is set by the remote terminal upon detection of an error in the message or upon detection of an invalid message (i.e. Illegal Command) to the terminal. The error may occur in any of the data words within the message. When the terminal detects an error and sets this bit, none of the data received within the message is used. In fact, once an error is detected, the terminal need not continue decoding the rest of the message, though most do.

If an error is detected within a message and the ME bit is set, the remote terminal must suppress the transmission of the status word. If the terminal detected an Illegal Command, the ME bit is set and the status word is transmitted. A logic “1” indicates an error. All remote terminals *must* implement the ME bit in the status word.

Instrumentation

The Instrumentation bit (bit time 10) is provided to differentiate between a command word and a status word (remember they both have the same sync pattern). The instrumentation bit in the status word is *always* set to logic

"0". If used, the corresponding bit in the command word is set to a logic "1". This bit in the command word is the most significant bit of the Subaddress field, and therefore, would limit the subaddresses used to 10000 - 11110, hence reducing the number of subaddresses available from 30 to 15. The instrumentation bit is also the reason why there are two mode code identifiers (00000B and 11111B), the latter required when the instrumentation bit is used.

Earlier monitoring systems required the use of the instrumentation bit to differentiate between command and status words. However, the price paid (loss of subaddresses) was too high for modern applications. Most monitors today are capable of following the protocol and message formats to determine which word is which.

Service Request

The Service Request bit (bit time 11) is provided so that the remote terminal can inform the bus controller that it needs to be serviced. This bit is set to a logic "1" by the subsystem to indicate that servicing is needed. This bit is typically used when the bus controller is "polling" terminals to determine if they require processing.

The bus controller, on receiving this bit set to a logic "1", takes a predetermined action such as issuing a series of messages or requests further data from the remote terminal. The later approach can be accomplished by requesting the terminal to transmit data from a defined Subaddress or by using the Transit Vector Word Mode Code.

Reserved

Bit times 12-14 are reserved for future growth of the standard and *must* be set to a logic "0". The bus controller should declare a message in error if the remote terminal responds with any of these bits set in its status word.

Broadcast Command Received

The Broadcast Command Received bit (bit time 15) indicates that the remote terminal received a valid broadcast command. On receiving a valid broadcast command, the remote terminal sets this bit to logic "1" and suppresses the transmission of its status words. The bus controller may issue a Transmit Status Word or Transmit Last Command Word Mode Code to determine if the terminal received the message properly.

Busy

The Busy bit (bit time 16) is provided as a feedback to the bus controller as to when the remote terminal is unable to move data between the remote terminal electronics and the subsystem in compliance to a command from the bus controller.

In the earlier days of 1553, the busy bit was required because many of the subsystem interfaces (analogs, synchros, etc.) were much slower compared to the speed of the multiplex data bus. Some terminals were not able to move the data fast enough. So instead of loosing data, a terminal was able to set the busy bit, indicating to the bus controller cannot handle new data at this time, and for the bus controller to try again later. As new systems have been developed, the need for the busy bit has been reduced. There are, however, still systems that need and have a valid use for the busy bit.

Examples of these are radios where the bus controller issues a command to the radio to tune to a certain frequency. It may take the radio several seconds to accomplish this, and while it is tuning, it may set the busy bit to inform the bus controller that it is doing as told. Another example is a tape or disk drive, which may take from milliseconds to seconds to store or retrieve a particular piece of data.

When a terminal is busy, it does not need to respond to commands in the "normal" way. For received commands, the terminal collects the data, but doesn't have to pass the data to the subsystem. For transmit commands, the terminal transmits its status word only. Therefore, while a terminal is busy, the data it supplies to the rest of the system is not available. This can have an overall effect upon the flow of data within the system and may increase the data latency within time critical systems (e.g., flight controls).

Some terminals used the busy bit to overcome design problems, setting the busy bit when needed. Notice 2 to the standard "strongly discourages" the use of the busy bit. However, there are still valid needs for its use. Therefore, if used, Notice 2 now requires that the busy bit may be set only as the result of a particular command received from the bus controller and *not* due to an internal periodic or processing function. By following this requirement, the bus controller, with prior knowledge of the remote terminal's characteristics, can determine what will cause a terminal to go busy and minimize the effects on data latency throughout the system.

Subsystem Flag

The Subsystem Flag bit (bit time 17) is used to provide "health" data regarding the subsystems to which the remote terminal is connected. Multiple subsystems may logically "OR" their bits together to form a composite health indicator. This single bit serves only as an indicator to the bus controller and user of the data that a fault or failure exists. Further

information regarding the nature of the failure must be obtained in some other fashion. Typically, a Subaddress is reserved for BIT information, with one or two words devoted to subsystem status data.

Dynamic Bus Control Acceptance Bit

The Dynamic Bus Control Acceptance bit (bit time 18) informs the bus controller that the remote terminal has received the Dynamic Bus Control Mode Code and has accepted control of the bus. For the remote terminal, the setting of this bit is controlled by the subsystem and is based on passing some level of built-in-test (i.e., a processor passing its power-up and continuous background tests).

The remote terminal, on transmitting its status word, becomes the bus controller. The bus controller, on receiving the status word from the remote terminal with this bit set, ceases to function as the bus controller and may become a remote terminal or bus monitor.

Terminal Flag

The Terminal Flag bit (bit time 19) informs the bus controller of a fault or failure within the remote terminal circuitry (only the remote terminal). A logic "1" shall indicate a fault condition.

This bit is used solely to inform the bus controller of a fault or failure. Further information regarding the nature of the failure must be obtained in some other fashion. Typically, a Subaddress (often 30) is reserved for BIT information, or the bus controller may issue a Transmit BIT Word Mode Code.

It should be noted that the standard states that this bit is for the remote terminal, meaning the entire remote terminal and not just the channel one which the command was received. In recent years, some component manufacturers have developed "chips" that are designed for single bus applications. Terminal designers have applied these chips to remote terminals indented for dual redundant applications. When these chips are used in these dual redundant applications, the status word, and subsequently the Terminal Flag bit, reflects only the status of the channel and not the entire remote terminal. (A dual redundant standby terminal has two channels: A and B). While not necessarily within the intent of the standard, there exist terminals that contain these chips and the bus controller needs to be aware of how the terminal defines the Terminal Flag (i.e. terminal or channel).

The last bit (bit time 20) is the word parity bit. Only odd parity is used.

CHAPTER 4

4. Message Formats

Introduction

The primary purpose of the data bus is to provide a common media for the exchange of data between systems. The exchange of data is based on message transmissions. The standard defines ten types of message transmission formats. All of these formats are based on the three word types just defined. The ten message formats are shown in Figure 7 and Figure 8. The message formats have been divided into two groups. These are referred to within the standard as the "information transfer formats" (Figure 7) and the "broadcast information transfer formats" (Figure 8).

The information transfer formats are based on the command/response philosophy in that all error free transmissions received by a remote terminal are followed by the transmission of a status word from the terminal to the bus controller. This handshaking principle validates the receipt of the message by the remote terminal.

Broadcast messages are transmitted to multiple remote terminals at the same time. The terminals suppress the transmission of their status words (not doing so would have multiple boxes trying to talk at the same time and thereby "jam" the bus). In order for the bus controller to determine if a terminal received the message, a polling sequence to each terminal must be initiated to collect the status words.

Each of the message formats is summarized in the sections, which follow.

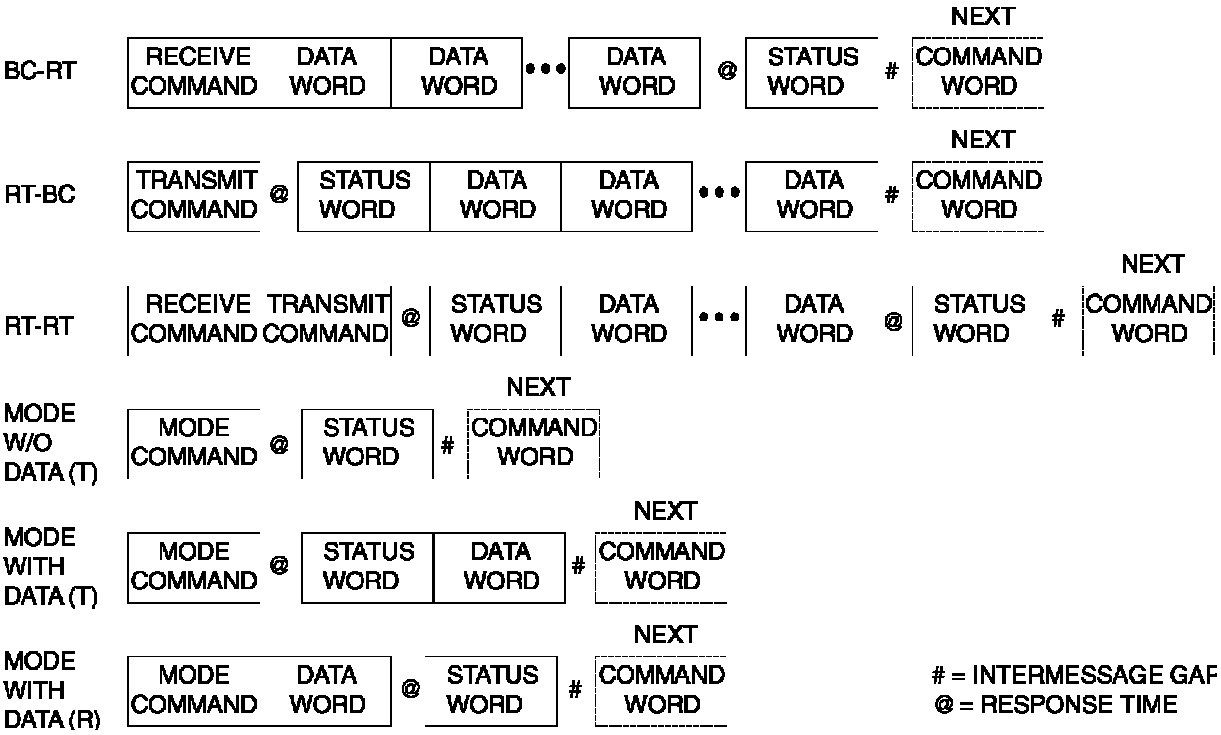

Figure 7. Information Transfer Formats

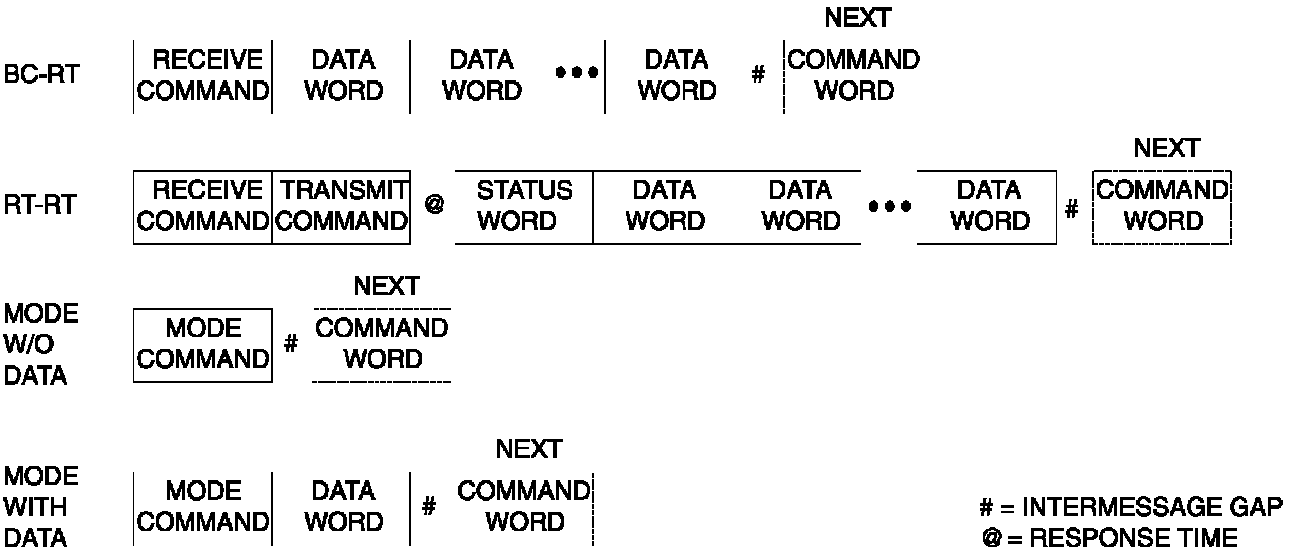

Figure 8. Information Transfer Formats (Broadcast)

Bus Controller to Remote Terminal

The bus controller to remote terminal (BC-RT) message is referred to as the *receive* command since the remote terminal is going to receive data. The bus controller outputs a command word to the terminal defining the Subaddress of the data and the number of data words it is sending. Immediately (without any gap in the transmission), the number of data words specified in the command word is sent.

The remote terminal upon validating the command word and all of the data words issues its status word within the response time requirements (maximum of 12 uSec).

The remote terminal must be capable of processing the next command that the bus controller issues. Therefore, the remote terminal has approximately 56 microseconds (status word response time [12 uSec] plus status word transmit time [20 uSec] plus inter-message gap (minimum 4 uSec) plus command word transmit time [20 uSec]) to either pass the data to the subsystem or buffer the data (see Data Buffering).

Remote Terminal to Bus Controller

The remote terminal to bus controller (RT-BC) message is referred to as a *transmit* command. The bus controller issues only a transmit command word to the remote terminal. The terminal, on validating the command word, transmits its status word followed by the number of data words requested by the command word.

The remote terminal doesn't know the sequence of commands to be sent and doesn't normally operate on a command until the command word has been validated. Therefore, it must be capable of receiving from the subsystem the data required within approximately 28 microseconds (the status word response time (12 uSec) plus the status word transmission time (20 uSec) minus an amount of time for message validation and transmission delays through the encoder and transceiver).

Remote Terminal to Remote Terminal

The remote terminal to remote terminal (RT-RT) command allows a terminal (the data source) to transfer data directly to another terminal (the data sink) without going through the bus controller. However, the bus controller may also collect the data and use them. (This is sometimes called a RT-RT-M command, since the Bus Controller monitors the data.)

The bus controller issues a command word to the receiving terminal immediately followed by a command word to the transmitting terminal. The receiving terminal is expecting data, but instead of data after the command word it sees a command sync (the second command word). The receiving terminal ignores this word and waits for a word with a data sync.

The transmitting terminal ignored the first command word (it did not contain his terminal address). The second word was addressed to it, so it processes the command as a RT-BC command by transmitting its status word followed by the required data words.

The receiving terminal, having ignored the second command word, again sees a command (status) sync on the next word and waits. The next word (the first data word sent) now has a data sync and the receiving remote terminal starts collecting data. After receipt of all of the data words (and validating), the terminal transmits its status word.

RT-RT Validation

There are several things that the receiving remote terminal of a RT-RT message should do. First, Notice 2 requires that the terminal time out in 54 to 60 microseconds after receipt of the command word. If the transmitting remote terminal didn't validate its command word (and no transmission occurred), the receiving terminal does not collect data from some new message. This could occur if the next message is either a transmit or receive message, where the terminal ignores all words with a command/status sync and would start collecting data words beginning with the first data sync. If the same number of data words was being transferred in the follow-on message, and the terminal did not test the command/status word contents, then the potential exists for the terminal to collect erroneous data.

The other function that the receiving terminal should do, but is not required by the standard, is to capture the second command word and the first transmitted data word. The terminal could compare the terminal address fields of both words to insure that the terminal transmitting was the one that was commanded to. This would allow the terminal to provide a level of protection for its data and subsystem. A systems-level question would occur on a terminal, subsystem, system, or aircraft if a terminal received data it was not suppose to receive (see Data Validity and Illegal Commands).

Mode Command Formats

Three mode command formats are provided. This allows for mode commands with no data words and for the mode commands with one data

word (either transmitted or received). The status/data sequencing is the same as the BC-RT or RT-BC messages except that the data word count is either one or zero. Mode codes and their use are described later.

Broadcast Information Transfer Formats

The broadcast information transfer formats, as shown in figure 8, is identical to the non-broadcast formats described above with the following two exceptions.

- The bus controller issues commands to terminal address 31 (11111B) that is reserved for this function.
- The remote terminals receiving the messages (those that implement the broadcast option) suppress the transmission of their status word.

The broadcast option can be used with the message formats where the remote terminal receives data. Obviously, multiple terminals cannot transmit data at the same time, so the RT-BC transfer format and the transmit mode code with data format cannot be used. The broadcast RT-RT allows the bus controller to instruct all remote terminals to receive and then instructs one terminal to transmit, thereby allowing a single subsystem to transfer its data directly to multiple users.

Notice 2 allows the bus controller to use only broadcast commands with mode codes (see Broadcast Mode Codes). Remote terminals are allowed to implement this option for all broadcast message formats. The Notice further states that the terminal must differentiate the subaddresses between broadcast and non-broadcast message (see Subaddress Utilization).

Command and Message Validation

The remote terminal must validate the command word and all data words received as part of the message. The criteria for a valid command word is:

- Word begins with a valid command sync.
- Valid terminal address (matches the assigned address of the terminal or the broadcast address if implemented).
- All bits are in a valid Manchester code.
- There are 16 information field bits, and there is a valid parity bit (odd).

The criterion for a data word is the same, except a valid data sync is required, and the terminal address field is not tested.

If a command word fails to meet the criteria, the command is ignored. After the command has been validated, and a data word fails to meet the

criteria, the terminal shall set the Message Error bit in the status word and suppress the transmission of the status word. Any single error within a message invalidates the entire message and the data are not used.

Illegal Commands

The standard allows remote terminals to monitoring for Illegal Commands. An Illegal Command is one that meets the valid criteria for a command word, but is a command (message) that isn't implemented by the terminal. An example would be a terminal that outputs only 04 data words to Subaddress 01 while a command word was received by the terminal that requested it to transmit 06 data words from Subaddress 03. This command, while still a valid command, could be considered by the terminal as illegal. The standard states only that the bus controller shall not issue illegal or invalid commands.

The standard provides the terminal designer with two options. First, the terminal can respond to all commands as usual (this is referred to as "responding in form"). The data received is typically placed in a series of memory locations that are not accessible by the subsystem or applications programs. This is referred to as the "bit bucket". All invalid commands are placed into the same bit bucket. For invalid transmit commands, the data transmitted is read from the bit bucket. Remember, the bus controller is not suppose to send these invalid commands, but you should beware anyway!

The second option is for the terminal to monitor for Illegal Commands. For most terminal designs, this is as simple as a look-up PROM with the T/R bit, Subaddress and word count fields supplying the address, and the output being a single bit that indicates if the command is valid or not. If a terminal implements Illegal Command detection and an illegal command are received, the terminal sets the Message Error bit in the status word and responds with the status word. There are good reasons for implementing this option and three examples are provided below.

Example 1

While the bus controller should never send illegal commands, the remote terminal should protect its data and applications software, especially if it is classified as flight or safety critical, such as flight controls or a pilot's display. If invalid data (data not intended for this terminal) were written into memory based on an Illegal Command, and subsequently processed by an applications program, it could have disastrous effects upon control laws within an autopilot. Also, it could provide erroneous data on a pilot's display causing the pilot to make wrong decisions.

Example 2

A second use for Illegal Command detection is to allow different commands to be received at different times. Notice 2 allows broadcast mode commands. While some of these commands help in system management (e.g., broadcast synchronize), others could have disastrous effects (e.g., broadcast reset). By providing an additional input into the look-up PROM, such as a weight-on-wheels signal, certain commands could be legal when on the ground, such as maintenance commands, or the broadcast reset or self-test, yet these commands could be illegal in flight.

Example 3

A third reason deals with testing. During initial development and integration, it is easy to make a programming error, which could change a single bit in the bus controllers command word. If a terminal responds in form, this error may take a long time to detect. However, if Illegal Command detection is implemented, whereas the terminal responds with the Message Error bit set in the status word, all 1553 test equipment catches this event and informs the operator that something isn't right, making testing and integration easier.

Terminal Response Time

The standard states that a remote terminal, upon validation of a transmit command word or a receive message (command word and all data words) shall transmit its status word to the bus controller. The response time is the amount of time the terminal has to transmit its status word. To allow for accurate measurements, the time frame is measured from the mid-crossing of the parity bit of the command word to the mid-crossing of the sync field of the status word. The minimum time is 4.0 microseconds; the maximum time is 12.0 microseconds. However, the actual amount of "dead time" on the bus is 2 to 10 microseconds, since half of the parity and sync waveforms are being transmitted during the measured time frame. It is important to remember this when calculating bus loading.

The standard also specifies that the bus controller must wait a minimum of 14.0 microseconds for a status word response before determining that a terminal has failed to respond. In applications where long data buses are used or where other special conditions exist, it may be necessary to extend this time to 20.0 microseconds or greater.

Finally, while the standard specifies that a remote terminal must respond within 12.0 microseconds with a valid status word, there are many remote terminal implementations that simply cannot reliably respond that quickly.

For this reason, most bus controllers have some method for adjusting the response time error threshold.

Inter-message Gap

The bus controller must provide for a minimum of 4.0 microseconds between messages. This time frame is measured from the mid-crossing of the parity bit of the last data word or the status word and the mid-crossing of the sync field of the next command word. The actual amount of "dead time" on the bus is 2 microseconds since half of the parity and sync waveforms are being transmitted during the measured time frame. Again, it is important to remember this when calculating bus loading.

The amount of time required by the bus controller to issue the next command is a function of the controller type (word, message, or frame). The gap typically associated with word controllers is between 40 and 100 microseconds. Message controllers typically can issue commands with a gap of 10 to 30 microseconds.

Many frame controllers are capable of issuing commands at the 4-microsecond rate. These controllers often require a time delay to slow them down, since there are many remote terminals (and even some bus monitors) that cannot keep up with this rate. In some cases, the type of command may influence the required inter-message gap. Broadcast commands to a device that is simulating multiple remote terminals may, for example, necessitate a longer gap while all of the simulated remote terminals process the message.

Superseding Commands

A remote terminal must always be capable of receiving a new command. This may occur when operating on a command on bus A, and after the minimum inter-message gap, a new command appears, or when operating on bus A, a new command appears on bus B. This is referred to as a Superseding command. A second valid command (the new command) shall cause the terminal to stop operating on the first command and start on the second. For dual redundant applications, this requirement implies that all terminals must, as a minimum, have two receivers, two decoders, and two sets of command word validation logic.

CHAPTER 5

5. Mode Codes

Introduction

Mode Codes are defined by the standard to provide the bus controller with data bus management and error handling/recovery capability. The mode codes are divided into two groups: those with, and those without, a data word.

The data word that is associated with the mode codes—and only one word per mode code is allowed—contains information pertinent to the control of the bus and does not generally contain information required by the subsystem (the exception may be the Synchronize with Data Word Mode Code). The mode codes are defined by bit times 15-19 of the command word. The most significant bit (bit 15) can be used to differentiate between the two-mode code groups. When a data word is associated with the mode code, the T/R bit determines if the data word is transmitted or received by the remote terminal. The mode codes are listed in Table 5.

Table 5. Mode Codes

T/R	Mode Code	Function	Data Word	Broadcast
1	00000	Dynamic Bus Control	No	No
1	00001	Synchronize	No	Yes
1	00010	Transmit Status Word	No	No
1	00011	Initiate Self-test	No	Yes
1	00100	Transmitter Shutdown	No	Yes
1	00101	Override Transmitter Shutdown	No	Yes
1	00110	Inhibit Terminal Flag Bit	No	Yes
1	00111	Override Inhibit Terminal Flag Bit	No	Yes
1	01000	Reset Remote Terminal	No	Yes
1	01001	Reserved	No	TBD
|	|	|	|	|
|	|	|	|	|
1	01111	Reserved	No	TBD
1	10000	Transmit Vector Word	Yes	No
0	10001	Synchronize	Yes	Yes
1	10010	Transmit Last Command Word	Yes	No
1	10011	Transmit BIT Word	Yes	No
0	10100	Selected Transmitter Shutdown	Yes	Yes
0	10101	Override Selected Transmitter Shutdown	Yes	Yes
1/0	10110	Reserved	Yes	TBD
|	|	|	|	|
|	|	|	|	|
1/0	11111	Reserved	Yes	TBD

Mode Code Identifier

The mode code identifier is contained in bits 10-14 of the command word. When this field is 00000B or 11111B, the contents of bits 15-19 are decoded as a mode code. Two mode code identifiers are provided so that the system can use the Instrumentation bit if desired.

The two mode code identifiers can't convey different information. In keeping with this requirement, many remote terminal implementations do not provide any indication to the subsystem of which Subaddress was specified in the actual mode code.

Dynamic Bus Control

The Dynamic Bus Control Mode Code is used to pass control of the data bus between terminals, thus supplying a “round robin” type of control.

Using this methodology, each terminal is responsible for collecting the data it needs from all other terminals. When it is finished collecting data, it passes control to the next terminal in line (based on some predefined sequence). This allows the applications program (the end user of the data) to collect the data when it needs them, always insuring that the data collected is from the latest source sample and has not been residing in a buffer waiting to be used.

Notices 1 and 2 to the standard forbid the use of Dynamic Bus Control for Air Force applications. Problems that may occur when a terminal, which is passed control, is unable to perform or does not properly forward the control to the next terminal, resulting in no one being in control. Control then has to be re-established by some terminal. The potential amount of time required re-establishing control could have disastrous effects upon the system (especially a flight control system).

A remote terminal that is capable of performing as the bus control should be capable of setting the Dynamic Bus Control Acceptance Bit in the terminal's status word to logic “1” when it receives the mode code command. Typically, the logic associated with setting this bit is based on the subsystem's (computer's) ability to pass some level of confidence test. If the confidence test is passed, then the bit is set and the status word is transmitted when terminal receives the mode command, thereby saying that it will assume the role of bus controller.

The bus controller can only issue the Dynamic Bus Control mode command to one remote terminal at a time. Obviously, the command is issued only to terminals that are capable of performing as a bus controller. Upon transmitting the command, the bus controller must check the terminal's status word to determine if the Dynamic Bus Control Acceptance Bit is set. If set, the bus controller ceases to function as the controller and becomes either a remote terminal or a bus monitor. If the bit in the status word is not set, the remote terminal which was issued the command is not capable of becoming the bus controller, and the current controller must remain the bus controller or attempt to pass the control to some other terminal.

Synchronize

The synchronize mode code is used to establish some form of timing between two or more terminals. This mode code does not use a data word. Therefore, the receipt of this command by a terminal must cause some predefined event to occur. Some examples of this event may be: the clearing, incrementing, or presetting of a counter; the toggling of an output signal; or the calling of some software routine. Typically, this command is used to time correlate a function such as the sampling of navigation data (present position, rates, etc.) for flight controls or targeting/fire control systems. Other uses have been for the bus controller to "sync" the backup controllers (or monitors) to the beginning of a major/minor frame processing.

When a remote terminal receives the Synchronize Mode Command, it should perform its predefined function.

For a bus controller, issuing of the command is all that is needed. The terminal's status word indicates only that the message was received, not that the "sync" function was performed.

Transmit Status Word

This is one of the two commands that *do not* cause the remote terminal to reset or clear its status word.

Upon receipt of this command, the remote terminal transmits the status word that was associated with the previous message, *not* the status word of the mode code message.

The bus controller uses this command for control and error management of the data bus. If the remote terminal had detected an error in the message and suppressed its status word (see Status Word - Message Error Bit), the bus controller can issue this command to the remote terminal to determine if the non-response was due to an error. As this command does not clear the status word from the previous message, a detected error by the remote terminal in a previous message is indicated by the Message Error bit being set in the status word.

The bus controller also uses this command when "polling". If a terminal doesn't have periodic messages, the RT can indicate when it needs communications by setting its Service Request bit in the status word. The bus controller, by requesting the terminal to transmit only its status word can determine if the terminal is in need of servicing and can subsequently issue the necessary commands. This "polling" methodology has the potential of reducing the amount of bus traffic by eliminating the transmission of unnecessary words.

The bus controller can also use this message to determine when a terminal is able to process data by examining the Busy bit in the status word. A terminal that is not able to transmit or receive data may set its Busy bit (the use of this is discouraged in Notice 2). Upon seeing this bit set, the bus controller may periodically "poll" (typically once per major frame) the terminal to determine when the terminal is ready for new commands (i.e., no longer busy).

This command is also used when broadcast message formats are used. As all of the remote terminals will suppress their status words, "polling" each terminal for its status word would reveal whether the terminal received the message by having its Broadcast Command Received bit set.

Initiate Self-test

This command, when received by the remote terminal, causes the remote terminal to enter into its self-test. This command is normally used as a ground based maintenance function, as part of the system power-on tests, or in flight as part of a fault recovery routine.

Note: This test is *only* for the *remote terminal*. Remember that the remote terminal consists of only the electronics necessary to communicate with the data bus, not the entire box, subsystem, computer, or memory.

In earlier applications, some remote terminals, on receiving this command, would enter self-test and go "off-line" for long periods of time. Notice 2, in an effort to control the amount of time that a terminal could be "off-line" limited the test time to 100.0 milliseconds following the transmission of the status word by the remote terminal.

While a terminal is performing a self-test, it may respond to a valid command in the following ways:

- No response on either bus ("off-line").
- Transmit only the status word with the Busy bit set.
- Normal response.

The remote terminal may, on receiving a valid command received after this mode code, terminate its self-test. As a subsequent command could abort the self-test, after issuing this command, the bus controller should suspend transmissions to the terminal for the specified amount of time (either a time specified for the remote terminal or the maximum time of 100.0 milliseconds).

Transmitter Shutdown

The bus controller uses this command to manage the bus. If a terminal's transmitter continuously transmits, this command provides a mechanism to turn the transmitter off. It is for dual redundant standby applications only.

Note: For a terminal's transmitter to be continuously outputting (this is often referred to as a babbling terminal), two errors (failures) have to occur within the terminal. First the logic that controlled the enable signal to the transmitter has to fail, and second, the terminal's fail-safe timer (maximum of 800.0 microseconds) has to have failed. In some designs that use a digital counter for the fail-safe timer, a single failure in a clock line could cause a babbling transmitter.

When it receives this command, the remote terminal shuts down (turns off) the transmitter associated with the opposite data bus. If a terminals transmitter is babbling on the A bus, the bus controller would send this command to the terminal on the B bus (a command on the A bus would not be received by the terminal).

The standard does not address what to do with the receiver associated with the transmitter being shut down. Terminal designs exist where the receiver is turned off; others exist where the receiver is left on. When the receiver is left on and the terminal is issued a command on the bus, the bus controller must be aware that the terminal could be processing this data without providing feedback (the status word) on the messages being processed and the messages being ignored because of an error.

Override Transmitter Shutdown

This command is the complement of the previous command in that it provides a mechanism to turn on a transmitter that has previously been turned off. When the remote terminal receives this command, it sets its control logic so that the transmitter associated with the opposite bus is allowed to transmit when a valid command is received on the opposite bus.

The only other command that can enable the transmitter is the Reset Remote Terminal Mode Command.

Inhibit Terminal Flag

This command controls of the Terminal Flag bit in a terminal's status word. The Terminal Flag bit indicates that there is an error within the remote terminal hardware (remember the definition), and that the data being transmitted or the data that was received may be in error. If this is

the case, then transmissions to that terminal should cease. However, the fault within the terminal may not have any effect upon the quality of the data, and the bus controller may elect to continue with the transmissions knowing a fault exists.

Within most bus controllers, a status word with the Terminal Flag bit set causes a special error processing routine to be called (either in the microcode or the software). If the bus controller is a computer, then this bit typically calls an interrupt service routine associated with error handling. While it is possible to mask this interrupt for a terminal, it becomes burdensome to have to mask and unmask on a terminal-by-terminal basis. If the bus controller is a microcoded state machine, then the masking of this bit internal to the bus controller may not be possible. Therefore, this mode command provides for "masking" of this bit at its source, the remote terminal.

The remote terminal receiving this command sets its Terminal Flag bit to logic "0" regardless of the true state of this signal. The standard does not state that the built-in-test that controls this bit be halted, but only the results are negated to "0".

Override Inhibit Terminal Flag

This command is the complement of the previous command in that it provides a mechanism to turn on the reporting of the Terminal Flag bit. When the remote terminal receives this command, it sets its control logic so that the Terminal Flag bit is properly reported based upon the results of the terminal's built-in-test functions.

The only other command that can enable the response of the Terminal Flag bit is the Reset Remote Terminal Mode Command.

Reset Remote Terminal

This command, when received by the remote terminal, causes the terminal electronics to reset to the power up state. Thus, if a transmitter had been disabled or the Terminal Flag bit inhibited, these functions would be reset as if the terminal had just powered up. Again remember that the reset applies only to the remote terminal electronics and not to the entire box.

Notice 2 restricts the amount of time that a remote terminal can take to reset its electronics. After transmission of its status word, the remote terminal resets within 5.0 milliseconds. While a terminal is resetting, it may respond to a valid command in one of the following ways:

- No response on either bus ("off-line").

- Transmit only the status word with the Busy bit set.
- Normal response.

The remote terminal may, upon receipt of a valid command received after this mode code, terminate its reset function. As a subsequent command could abort the reset, the bus controller after issuing the command, should suspend transmissions to the terminal for the specified amount of time (either a time specified for the remote terminal or the maximum time of 5.0 milliseconds).

Transmit Vector Word

This command causes the remote terminal to transmit a data word referred to as the vector word. The vector word identifies to the bus controller service request information relating to the message needs of the remote terminal. While not required, this mode code is often tied to the Service Request bit in the status word.

The contents of the data word inform the bus controller of messages needed to be sent. While there are no requirements in the standard regarding the contents of this data word, Section 80 of MIL-HDBK-1553A offers two examples. The first is a coded field, which the bus controller must decode to determine the requirements. This requires prior knowledge of all requirements by the bus controller and makes changes to the system difficult because both the remote terminal and the bus controller (and all the backup controllers) must be updated.

The second example is perhaps the easiest and most elegant. The data word (vector word) contains the command word that the terminal requires be sent by the bus controller. The bus controller need only echo this word as a command. This requires no special decoding by the controller and makes changes easier in that only the remote terminal is modified.

The bus controller also uses this command when “polling”. Typically, this command is used in conjunction with the Service Request bit in the status word. The bus controller requests only the status word (Transmit Status Word Mode Code). On seeing the Service Request bit set, it issues the Transmit Vector Word Mode Code. The bus controller can always ask for the Vector Word (always getting the status word anyway) and reduce the amount of time required to respond to the terminal's request.

Synchronize with Data Word

The purpose of this synchronize command is the same as the synchronize without data word, except this mode code provides a data word to provide additional information to the remote terminal. The contents of the data

word are left to you. Examples from "real world" applications have used this word to:

- Provide the remote terminal with a counter or clock value.
- Provide a backup controller with a frame identification number (minor frame or cycle number).
- Provide a terminal with a new base address pointer used in extending the Subaddress capability (see Extended Sub addressing).
- Provide a monitoring system with the minor frame number.

Transmit Last Command Word

This is one of the two commands that *do not* cause the remote terminal to reset or clear its status word.

Upon receiving this command, the remote terminal transmits the status word that was associated with the previous message and the Last Command Word (valid) that it received.

The bus controller uses this command for control and error management of the data bus. When a remote terminal is not responding properly, the bus controller can determine the last valid command the terminal received and can re-issue subsequent messages as required.

Transmit Built-in-Test (BIT) Word

This mode command is used to provide detail about the Built-in-Test status of the remote terminal. Its contents provide information regarding the remote terminal only (remember the definition) and not the subsystem.

While most applications associate this command with the Initiate Self-test Mode Code, the standard requires no such association. Typically, this command is used to issue the Initiate Self-test Mode Code, allow the required amount of time for the terminal to complete its tests, and then issue the Transmit BIT Word Mode Code to collect the results of the test.

Other applications have updated the BIT word on a periodic rate based on the results of a continuous background test (e.g., as data wrap-around test performed with every data transmission). This word can then be transmitted to the bus controller, upon request, without having to initiate the test and then wait for the test to be completed. The contents of the data word are left to the terminal designer.

Receipt by the remote terminal of either the Transmit Status Word Mode Code or the Transmit Last Command Word Mode Code doesn't alter the contents of the BIT data word.

Notice 2 to the standard states that any terminal which is capable of performing a built-in-test or some level of self-test must implement this mode code to provide the results of those tests.

Selected Transmitter Shutdown

Like the Transmitter Shutdown Mode Code, this mode code is used to turn off a babbling transmitter. The difference between the two mode codes is that this mode code has a data word associated with it. The content of the data word specifies which data bus (transmitter) to shutdown. This command is used in systems that provide more than dual redundancy, typically either tri- or quad- redundant systems associated with flight controls.

Override Selected Transmitter Shutdown

This command is the complement of the previous one in that it provides a mechanism to turn on a transmitter that had previously been turned off. When the remote terminal receives this command, the data word specifies the data bus (transmitter) that sets its control logic so that the transmitter associated with that bus be allowed to transmit when a valid command is received on that bus.

The only other command that can enable the selected transmitter is the Reset Remote Terminal Mode Command.

Reserved Mode Codes

As can be seen from the Mode Code table above, there are several bit combinations that are set aside as reserved. These are reserved for future growth. It should also be noted from the table that certain bit combinations are not listed. The standard allows the remote terminal to respond to these reserved and "undefined" mode codes in the following manner

- Set the message error bit and respond (see Illegal Commands).
- Respond in form.

As the designer of terminal hardware or a multiplex system, you are forbidden to use the reserved mode codes for any purpose.

Required Mode Codes

Notice 2 to the standard requires that all remote terminals implement the following four mode codes:

- Transmit Status Word.
- Transmitter Shutdown.
- Override Transmitter Shutdown.
- Reset Remote Terminal.

This requirement was levied so as to provide the multiplex system designer and the bus controller with a minimum set of commands for managing the multiplex system.

The above requirement was placed on the remote terminal. Notice 2 also requires that a bus controller be capable of implementing the entire mode codes; however, never used the Dynamic Bus Control Mode Code for Air Force applications.

Broadcast Mode Codes

Notice 2 to the standard allows broadcasting of mode codes (see the Mode Code table). Using the broadcast option can be of great assistance in the area of terminal synchronization. Ground maintenance and troubleshooting can take advantage of broadcast Reset Remote Terminal or Initiate Self-test, but these two commands can have disastrous effects if used while in flight. You must provide checks to insure that commands such as these aren't issued by the bus controller or operated upon by a remote terminal when certain conditions exists (e.g., in flight) (see Illegal Commands).

6. System Issues

We have now covered the hardware and protocol elements of the standard, but how do we use it? This is where the standard provides very little guidance and we must learn from real world applications. This section answers some of the systems-level questions that are most asked or appear to be of the greatest concern to new users of the standard.

Using Subaddresses

The standard provides no guidance on using subaddresses. Assignment of subaddresses and their functions (the data content) is left to you. While most designers automatically start assigning subaddresses with 01 and count upwards, it is recommended that the first series of Subaddress start at 16 and count up. After using the "top half" of the subaddress assignments, then use 01 through 15. Assigning subaddresses in this manner allows you to use the Instrumentation bit.

Consistency of Subaddress usage is a plus in any system. For example, if Subaddress 30 is reserved, system wide, for Built In Test (BIT) data, and Subaddress 20 is reserved, again system wide, for wrap-around messages, anytime these messages are recorded by a monitoring system their meaning is clear. Data messages may be distinguished from housekeeping and control messages by similar segmentation of subaddresses.

The standard also requires that normal subaddresses be separated from broadcast subaddresses. If the broadcast option is implemented, then an addition memory block is required for the receive broadcast commands.

Extended Subaddressing

The number of subaddresses that a terminal has is limited to 60 (30 transmit and 30 receive). Therefore, the number of unique data words available to a terminal is 1920 (60 x 32). For earlier applications, where data being transferred was analog sensor data and switch settings, this was

more than sufficient. However, in some of today's applications, with digital computers exchanging data or a video sensor passing digitized video data, the number of words is limiting.

Most terminal designs establish a block of memory for use by the 1553 interface circuitry. This block contains an address start pointer. Then the memory is offset by the Subaddress number and the word count number to arrive at a particular memory address.

A method of extending the range of the subaddresses has been successfully employed. This method uses either a dedicated Subaddress and data word or makes use of the synchronize with data word mode code. The data word associated with either of these contains an address pointer, which is used to re-establish the starting address of the memory block. Changing the blocks is controlled by the bus controller and can be done based on numerous functions. Take, for instance, operational modes, where one block is used for startup messages, a different block for take-off and landing, a different block for navigation and cruise, a different block for mission functions (i.e. attack or evade modes), and a different block maintenance functions.

Changing the start address can also be associated with minor frame cycles. Eight minor frames can have a separate memory block for each frame. The bus controller can synchronize frames and change memory pointers at the beginning of each new minor frame.

Computers exchanging large amounts of data (e.g., GPS Almanac Tables), or computers, which receive program loads through the data bus at power-up, could set the pointers at the beginning of a message block. Then they could send thirty 32 word messages, move the memory pointer to the last location in the remote terminals memory that received data, then send the next block of thirty 32 word messages. They could continue this cycle until the memory is loaded. The use is left up to you.

Data Wrap-Around

Notice 2 requires that terminal to perform a data wrap-around. Subaddress 30 is *suggested* for this function. Data wrap-around provides the bus controller with a method of testing the data bus from its internal circuitry, through the bus media, to the terminals internal circuitry. The bus controller sends the remote terminal a message block and then commands the terminal to send it back. The bus controller can compares the data sent with that received to determine the state of the data link. There are no special requirements on the bit patterns of the data being transferred.

The only design requirements are on the remote terminal. For the data wrap-around function, the terminal must be capable of sending the number of data words equal to the largest number of data word sent for any

transmit command. This means that if a terminal maximum data transmission is only four data words, it need only provide for four data words in its data wrap-around function.

The other requirement is that the remote terminal holds the data only until the next message. The normal sequence is for the bus controller to send the data, then ask for them back in the next message. If another message is received by the remote terminal before the bus controller requests the data, the terminal can discard the data from the wrap-around message and operate on the new command.

Data Buffering

The standard specifies that any error within a message invalidates the entire message. This implies that the remote terminal must store the data within a message buffer until the last data word has been received and validated before allow the subsystem access to the data. To insure that the subsystem always has the last message of valid data received to work with would require the remote terminal to, as a minimum, double buffer the received data.

There are several methods to accomplish this in hardware. One method is for the terminal electronics to contain a First-in First-out (FIFO) memory that stores the data as they are received. On validating the last data word, the terminals subsystem interface logic moves the contents of the FIFO into memory accessible by the subsystem. If an error occurs during the message, the FIFO is reset.

A second method establishes two memory blocks for each message in common memory. The subsystem is directed to read from one block (block A) while the terminal electronics writes to the other (Block B). Upon receipt of a valid message, the terminal switch pointers, indicating that the subsystem is to read from the new memory block (Block B) while the terminal now writes to block B. If an error occurs within the message, the memory blocks aren't switched.

Some of the off-the-shelf components available provide for data buffering. Most provide for double buffering, while some provided for multi levels of buffering.

Variable Message Blocks

Remote terminals should be able to transmit any subset of any message. This means that if a terminal has a transmit message at Subaddress 04 of 30 data words, it should be capable of transmitting any number of those data words (01-29) if commanded by the bus controller. The order in which the subset is transmitted should be the same as if the entire message is being transmitted (the content of data word 01 is the same regardless of the word count). This hasn't always been the case.

Terminals that implement Illegal Command detection might not consider subsets of a message illegal. If in our example above, a command is received for 10 data words. This might not be illegal. But, if a command were received for 32 data words, this would be considered an illegal command. The exact implementation depends on the commands that are expected. In some cases there might be limits, both upper and lower, on the number of words which may be included in a valid command.

Sample Consistency

When transmitting data, the remote terminal needs to ensure that each message transmitted is of the same sample set and contains mutually consistent data. Multiple words used to transfer multiple precision parameters or functionally related data must be from the same sampling.

If a terminal is transmitting pitch, roll, and yaw rates, and while transmitting, the subsystem updates this data in memory, but this occurs after pitch and roll have been read by the terminal's electronics, then the yaw rate transmitted would be of a different sample set. Having data from different sample rates could have undesirable effects.

Thus, the terminal must provide some level of buffering (the reverse of what was described above) or some level of control logic to block the subsystem from updating data while they are being read by the remote terminal.

Data Validation

The standard tightly defines the criteria for the validation of a message. All words must meet certain checks (e.g., valid sync, Manchester encoding, number of bits, odd parity) for each word and each message to be valid.

But what about the contents of the data word? MIL-STD-1553 provides the checks to insure the quality of the data transmission from terminal to

terminal, but is not responsible for the validation tests of the data. This is not the responsibility of the terminal electronics, but of the subsystem. If bad data are sent, then "garbage in equals garbage out". But the standard does not prevent you from providing additional levels of protection. The same techniques used in digital computer interfaces (i.e. disk drives, serial interfaces, etc.) can be applied to 1553. These techniques include checksums, CRC words, and error detection/correction codes. Section 80 of MIL-HDBK-1553A which covers data word formats even offers some examples of these techniques.

But what about using the simple indicators embedded within the standard. Each remote terminal provides a status word indicating not only the health of the remote terminal's electronics, but also that of the subsystem. However, in most designs, the status word is kept within the terminal electronics and not passed to the subsystems. In some off-the-shelf components, the status word is not even available to be sent to the subsystem if you wanted to. But two bits from the status word should be made available to the subsystem and the user of the data. These are the Subsystem Flag and the Terminal Flag bits.

What if a terminal received 32 words of data from another terminal (e.g., RT-RT transfer)? And these 32 words pass all of the 1553 tests (all bits and words are valid), but the subsystem flag bit is set. The source of the data might be telling you something is wrong. The source could be having memory problems, but the bus controller told the terminal to send data and here they are. The 1553 circuitry passed the data correctly, but they were bad to begin with. Now what? It isn't the responsibility of the terminal electronics to examine and pass judgment on the data, only to move them from box A to box B, insuring that the data out is the same as data in. The terminal should provide the subsystem all of the information regarding the data (e.g., the subsystem and terminal flag bits) so that it can determine whether or not to use the data.

Major/Minor Frame Timing

The standard specifies the composition of words (command, data, and status) and messages (information formats and broadcast formats). It provides a series of management messages (mode codes), but it does not provide any guidance on how to apply these within a system.

A remote terminal, based upon the contents of its data, typically states how often data are collected and the fastest rate it should be output. For input data, the terminal often states how often it needs certain data either to perform its job or maintain a certain level of accuracy. The rates are referred to as the transmission and update rates. It is your job to examine the data needs of all of systems and determine when data is transferred from whom to whom. This data is subdivided into periodic messages (those which must be transferred at some fixed rate) and aperiodic

messages (those which are typically either event driven – the operator pushes a button – or data-driven – a value is now with range).

A major frame is defined so that all periodic messages are transferred at least once, as defined by the message with the slowest transmission rate. Typical major frame rates used in today's applications vary from 40 to 640 milliseconds. There are some systems that have major frame rates in the 1 to 5 second ranges, but these are the exceptions. Minor frames are then established to meet the requirements of the higher update rate messages. Minor frame rates are usually based on a binary expansion of the major frame. Figure 9 is an example of the major/minor frame relationships.

Within the bus controller, the minor frame rate is set to provide a real time clock interrupt to the bus controller software. At the beginning of each new minor frame, an interrupt occurs, and the bus controller starts issuing the messages for that frame.

The sequence of messages within a minor frame is left undefined. Figure 9 shows two examples of how this can be done. In the first example, the bus controller starts the frame with a synchronize command to the back-up controller and any remote terminals that may require it. The frame synchronization is followed by the transmission of all of the periodic messages to be transferred in that minor frame. At the end of the periodic messages, the bus controller is either finished (resulting in dead bus time [no transmissions]) until the beginning of the next frame, or the bus controller can use this time to transfer aperiodic messages, error handling messages, or transfer data to the backup bus controller(s).

In the second example (typically used in a centralized processing architecture), the bus controller issues all periodic and aperiodic transmit messages (collects the data), then processes this data (possibly has dead time during this processing), and then issues all the receive messages (outputting the results of the processing). Both methods have been used successfully.

A third example is a system in which the bus controller communicates with a particular remote terminal only when it needs to perform a given task. There are no minor or major frames since there is no periodic message traffic. All of the 1553 traffic is the result of some non-periodic stimulus, perhaps the result of some operator inputs. This is the simplest case since the host processor software determines the next message to be transacted on the bus.

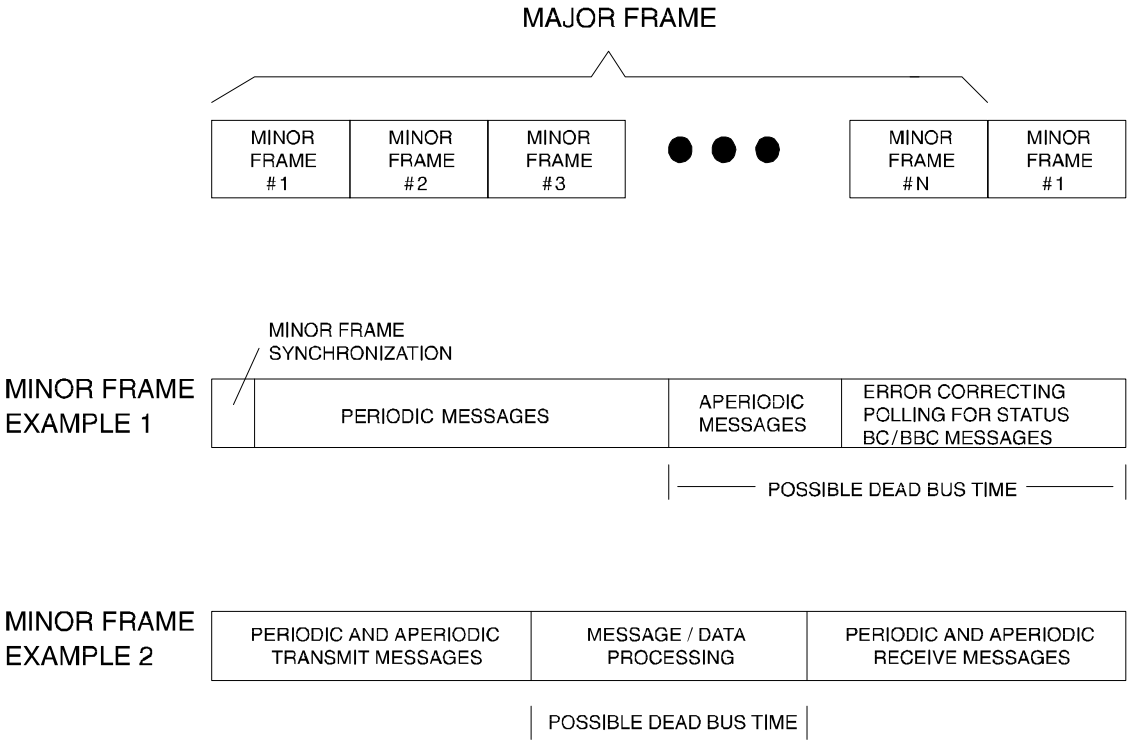

Figure 9. Major/Minor Cycles

Bus Loading

Bus loading is the percentage of time required for the messages to be sent (amount of time bus is busy) versus the total amount of time available (either the major frame time or normalized to one second). The numbers used for the calculation of all of the message formats can be found in MIL-HDBK-1553A. A simple summary is provided in Table 6.

Table 6. Bus Loading Numbers

Command Word	20 µSec
Status Word	20 µSec
Data Word(s)	20 µSec (each)
Inter-Message Gap	2 µSec minimum
Response Time	10 µSec maximum

Therefore, the amount of time for a 28 word receive command being issued by a bus controller with a 50 uSec inter-message gap is 658 uSec. If this were the only message in a 5-millisecond frame, then the loading would be 13 percent.

When calculating loading, be sure to use the actual response times and the actual inter-message gap times as these can vary from the values given in Table 6.

The amount of bus loading and the distribution of messages throughout the major frame have a major impact on the data throughput. The maximum practical data throughput is approximately 46,000 words per second, taking into account the overhead of the command and status words and the response and inter-message gap times, and assuming 32 data words per message. Shorter messages transfer fewer data words due to the overhead of command and status words, and the various gaps.

Most procurement specifications today state that 100 percent growth is required for the multiplex system and that any minor frame loading does not exceed 70 percent. A 100 percent growth means that major frame loading cannot exceed 50 percent. MIL-HDBK-1553A recommends that a target goal of 40 percent loading be used at system design and 60 percent by fielding. The extra time is used for system growth (adding new systems which was one of the reasons for going to a multiplex system to begin with) and for error recovery messages.

Error Processing

The amount and level of error processing is left to the systems designer but may be driven by the performance requirements of the system. Error processing is only afforded to critical messages, wherein the non-critical messages just wait for the next normal transmission cycle.

If a data bus is 60 percent loaded and each message received an error, error processing would exceed the 100 percent of available time and thereby cause problems within the system.

Error processing is a function of the level of sophistication of the bus controller. Some controllers (typically message or frame controllers) can automatically perform some degree of error processing. This usually is limited to a retransmission of the message either once on the same bus or once on the opposite bus. Should the retried message also fail, the bus controller software is informed of the problem. The message may then be retired at the end of the normal message lists for the minor frame.

If the error continues, it may be necessary to stop communicating with the terminal, especially if the bus controller is spending a large amount of time performing error processing. Some systems try to communicate with a terminal for a predefined number of times on each bus (typically 8). After this, all messages to the terminal are removed from the minor frame lists and substituted with a single transmit status word mode code. If the terminal starts responding, and passes some other level of confidence testing, the controller may reinstate the normal messages.

An analysis needs to be performed on the critical messages to determine the effects upon the system if they are not transmitted or the effects of data latency if they are delayed to the end of the frame.

7. Connecting the Bus

Introduction

Connecting a system of MIL-STD-1553 terminals isn't necessarily a trivial task. There are three options supported by the standard:

- Direct Coupled.
- Transformer Coupled.
- Some stubs Direct Coupled, other stubs Transformer Coupled.

Any 1553 bus consists of a single twisted-shielded-pair of wires, terminated at each end with proper termination resistors. Connections between terminals and the bus are made through the use of *stubs*. There are two types of stub, the Direct Coupled (or Short) stub and the Transformer Coupled (or Long) stub.

A Transformer or Long stub can be up to 20 feet in length, and typically connects to a small box or "coupler" which is wired in series with the actual bus.

A Direct or Short stub can be up to one foot long and is typically created by the use of a "Tee". The bus goes into and out of the Tee while the terminal is connected directly to the third leg of the Tee.

This chapter describes the most common connection schemes, and attempts to point out the most common pitfalls.

Direct Coupled Bus

Figure 10 shows how a single terminal can be connected to a bus controller using direct coupling. The bus is terminated at each end with a terminating resistor, and that the connection from the Tee to the terminal is very short. Typically the Tee is connected directly to the connector on the back of the terminal.

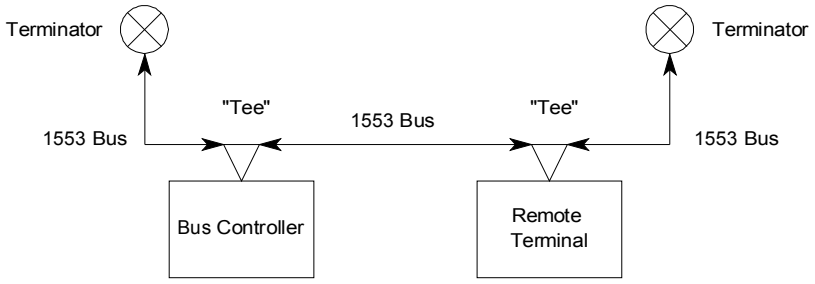

Figure 10. Direct Coupling

Transformer Coupled Bus

Figure 11 shows how a single terminal can be connected to a bus controller using transformer coupling. The bus is terminated at each end with a terminating resistor, just as in Figure 10, but now the two Tee's have been replaced with transformer couplers. The connection to the terminals is no longer the length of the Tee, but significantly long, consisting of another cable (of up to 20 feet in length).

Just because this coupling method is called “Long Stub” doesn’t mean that the stub needs to be 20 feet long. That is just the *maximum* length specified in the standard.

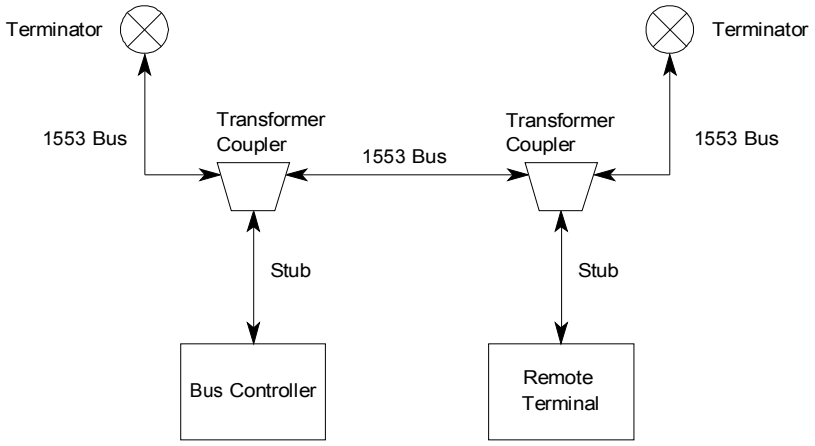

Figure 11. Transformer Coupling

Mixed Bus Coupling

In the example in Figure 12, the Bus Controller is connected to the bus using direct coupling and the remote terminal is connected using transformer coupling.

This method can be used when it is convenient to route the bus close to one of the terminals but it is not convenient to route it close to the other terminal.

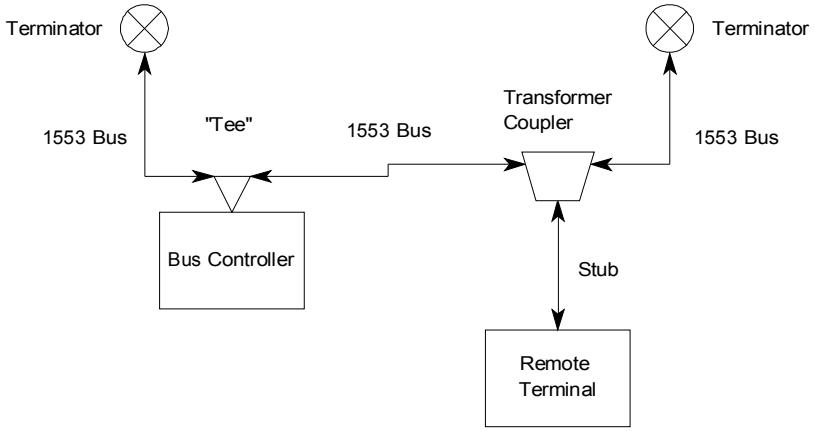

Figure 12. Mixed Bus Coupling

Primary and Secondary Buses

Typically, a Bus Controller, a Remote Terminal and a Bus Monitor will have *two* bus connectors, labeled variously "A, B", "Primary, Secondary", "Primary, Backup", "0, 1" etc.

How should these connectors be wired? Perhaps together?

Most 1553 hardware which is produced "off the shelf" today contains provisions for supporting a "*dual redundant*" 1553 bus. Dual Redundant is two complete 1553 buses, with only one of the buses active at a time. The bus controller chooses which bus to transact a given message on and the addressed remote terminal responds on that bus with the required data or status. The next message may be on the same bus, or on the other bus, as determined by the bus controller.

This means that *both* buses must be connected to *each* of the terminals in the system. And that the two buses must *not* be connected to each other.

This is illustrated in Figure 13 for transformer coupling.

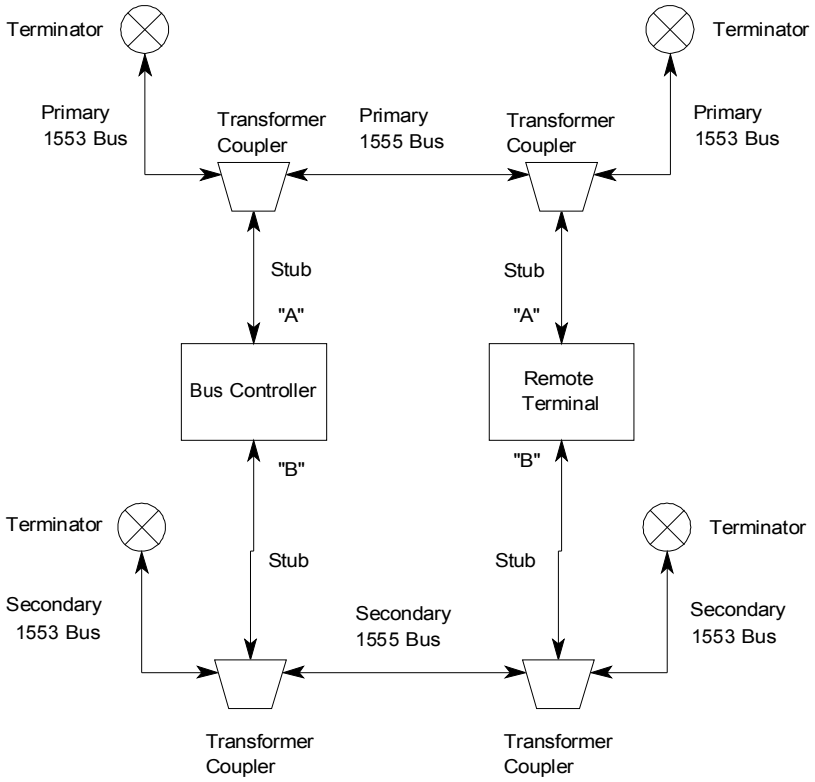

Figure 13. Primary and Secondary Buses

This normally requires that the bus wiring described above be duplicated for the two buses; there is two of everything. This is, of course, the definition of *dual redundant*. If the bus controller never accesses one of the buses, the wiring for that bus need not be supplied. This is sometimes done in some testing environments to reduce the cost of the test equipment. The disadvantage is that the second bus cannot be tested easily.

More than Two Terminals

When adding terminals to the above systems simply add more of the same parts.

To add a terminal using transformer coupling you need to add another transformer coupler in series with the bus (or use a coupler containing more than one transformer). Connect the stub connection on the transformer to the terminal with a cable. Then set the terminal to "Transformer" coupling. Do the same thing for the "B" bus and the job is complete.

To add a terminal using direct coupling you need to add another Tee in series with the bus. Connect the free end of the Tee to the back of the terminal. Then set the terminal to "Direct" coupling. Do the same thing for the "B" bus and the job is complete.

Cautions

Any standard that is useful is capable of being abused. Witness RS-232 connections; nothing in the RS-232 standard encourages the use of LED's on an RS-232 connector for determining the status of the connections, but such devices are in common use everyday. The parallel printer port on a PC is not intended for use as a source of power, but many devices "steal" their operating power from that source.

Since MIL-STD-1553 has spawned a great many useful system implementations, it stands to reason that some clever individuals might have come up with "extensions" to the standard which, while not "per the specs", are none the less useful.

While none of these tricks are being recommended, in some cases they have been used, sometimes even successfully, on various applications. In most cases they were *expected* to work, and when they didn't, they spawned a call to Technical Support.

You need to understand the effects of these variations on overall system performance and on the system's ability to evaluate accurately the validity of the use (or misuse) of the standard.

The Single Bus Wiring Short Cut

This short cut is simply to omit the second redundant bus wiring. The "B" or "Secondary" bus connections are simply not connected. This defeats the dual redundant nature of the typical terminal, but saves the additional wiring and couplers.

This trick is typically used in a lab environment during early integration and testing of a system employing 1553 buses, when it is not yet necessary to test the performance of the system while using the backup bus.

When using this trick, don't be surprised when communications on the "B" bus fail!

The Crossed Bus Wiring Short Cut

This trick never works! The idea is to connect both redundant buses, to avoid having to supply the full second set of wiring. This is typically implemented when the "Single Bus Wiring Short Cut" demonstrates that communications are not possible on the "B" bus, and someone needs that communication "right away". Often, someone attempts to install a jumper at one terminal that connects its "A" and "B" buses.

Due to the nature of their implementations, most terminals act badly when presented with concurrent data on both buses. They generate strange errors and in general fail to communicate correctly on either bus.

The Two Bus/One Controller Short Cut

One short cut is sometimes employed in test situations is to connect the "A" bus from the bus controller to some number of terminals, in the normal manner described above. The "B" bus from the bus controller is then connected to some other collection of terminals, effectively giving the bus controller access to twice the number of terminals, at the expense of loosing the dual redundant nature of the "standard" 1553 bus.

In such a system a terminal on the "A" bus can't communicate to a terminal on the "B" bus (for example, via a RT to RT message), but this is known and accepted before hand.

The Very Long Stub Short Cut

This trick consists of using a transformer coupled stub which exceeds 20 feet in length, or a direct coupled stub which exceeds one foot in length.

While there are cases on record of transformer coupled stubs over 100 feet long being used successfully, there is no guarantee that they will work, especially if there is something else wrong (e.g., not per the specification) on the bus.

Mixing direct and transformer coupling on the same bus also tends to reduce the length of stub that can be used successfully.

The No Terminators Short Cut

The idea is to omit both the couplers and the termination resistors and connect a bus controller directly (with a single cable) to a remote terminal. There are only two cases where this trick would be tried:

- The new hardware just came in; you want to try it, but you don't have the proper cables, terminators and/or couplers.
- You don't know any better.

One problem with this trick is that occassionally it works, leading one to believe that it will *always* work, when in fact it *hardly ever* works.

The Connected Couplers Short Cut

This trick attempts to connect two buses by connecting two transformer-coupled stubs. It is thought that communications are then possible between the two buses. It is most likely to be tried during a debug session.

This hardly ever works because of two problems:

- The addresses on the two buses overlap causing two terminals to have the same address on the now-joined bus.
- The voltage levels have been reduced by passing through two couplers to the point where signaling, if possible at all, is spotty and erratic.

Remember when thinking about using this trick that transformer couplers provide isolation (read “attenuation”) because of the internal resistors.

The Junk Wiring Short Cut

This trick involves cobbling the terminals together with whatever wiring is available, ignoring the shielding, grounding one side of the bus, and anything else which is not per the standard.

While sometimes these tricks work for a while, communications integrity is normally compromised resulting, sooner or later, in a call to Technical Support.

8. Testing

Introduction

Testing of a MIL-STD-1553 terminal or system is a not a trivial task. There are a large number of options available. A few of these include message formats, mode commands, status word bits, and coupling methodology. In addition, history has shown that different component manufacturers and designers have made different interpretations regarding the standard, thereby introducing products that implement the same function quite differently. Thus, coming up with a standard terminal is a near impossible feat.

Government and SAE Test Plans

For years, the Air Force provided for testing of MIL-STD-1553 terminals and components. Today, this testing is the responsibility of industry. The SAE, in conjunction with the government, has developed a series of Test Plans for all 1553 elements. These Test Plans are listed in Table 7.

Table 7. Test Plans

MIL-HDBK-1553	Remote Terminal Validation Test Plan Notice 1
MIL-HDBK-1553A	Remote Terminal Validation Test Plan Section 100
SAE AS-4112	Remote Terminal Production Test Plan
SAE AS-4113	Bus Controller Validation Test Plan
SAE AS-4114	Bus Controller Production Test Plan
SAE AS-4115	Data Bus System Test Plan
SAE AS-4116	Bus Monitor Test Plan
SAE AS-4117	Bus Components Test Plan

Test Equipment

ZhengHong Aviation Tech Co.,ltd provides product solutions to fill a great majority of 1553 marketplace needs. Such as the MIL-STD-1553 (A, B) test and simulation board card products, 1553 interface cards, MIL-STD-1553 (A, B) interface module, and 1553B bus tester.

Supported computer bus platform: ISA, PC104, PCI, CPCI, PXI, VME, USB, Ethernet etc.
Single function or multi function,1 or 2 channel
Notes: single function: operate as a Bus controller, multiple (up to 32) Remote Terminals or Bus Monitor;
Multi function: operate as Bus controller, multiple (up to 32) Remote Terminals and Bus Monitor.

Supported Operating Systems: Windows98/2000/xp/Linux/Vxworks
Set as MIL-STD-1553A or MIL-STD-1553B
speed supported 1M, 4M; 10M (pre-study in)
BC Automatic retry
BC Support for frame retransmission
Software to set the frame interval and the message interval time
Large-capacity data storage: 16Mx16bit
Dual redundant channel data to send and receive
8-channel TTL digital inputs and outputs
Driver: provides a standard DLL, supports VC, VB, Delphi, LabVIEW, CVI and other standard development language.
Applications: just install the 1553 card can be used to achieve most applications operating functions required for communication

Technical Specifications:
Power Requirements: +5VDC, 12VDC
Operating temperature: -40 to +85 deg C
Physical Dimensions: CPCI as a 3U,6U, other boards for the standard model
Connector: PC/104 bus with double the non-pin connector, other optional SCSI36, SCSI68 and so on, comes standard with a corresponding connector joints.

MIL-STD-1553 Accessories

ZhengHong Aviation Tech Co.,ltd also provide a full line of accessories for the MIL-STD-1553 data bus that include, 1553 Cables(M17/176-00002), 1553 Box / Inline Couplers,1553 Repeaters, 1553 Bus Switches, 1553 Break out boxes, 1553 Connectors, 1553 Terminators, 1553 Transformers and custom cable assemblies. From TYCO/Raychem, North hills, Milestek, Trompeter, Excalibur systems etc.

For more information on these products, please contact ZHHKTECH,
E-mail: support@zhhktech.com WEB: www.zhhktech.com

CHAPTER 9

9. Additional Information

Sources of Information

In addition to the test plans listed in Table 7, there are additional sources that can provide a great deal of insight and assistance in designing with MIL-STD-1553.

- **MIL-HDBK-1553A –** This government publication provides an explanation of the standard and a comparison of all versions of the standard. Guidance on media, terminal, and systems designs, along with current "real world" applications are included. The data word standard formats are also contained in this document.
- **AE-12 Multiplex Systems Integration Handbook** – This SAE publication is a series of papers presented on applications, interpretations, and lessons learned.
- **SAE 1553 Users Group** – This is a collection of industry and military experts in 1553 who provide an open forum for information exchange, and provide guidance and interpretations/clarifications with regards to the standard. This group meets twice a year.