

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Find the *Curtiss-Wright / Systran / Digital Technology DTI-1121* at our website: *Click HERE*

DTI-112X

User Manual
for the PC-AT

Document No. T-T-MU-112X####-A-0-A2

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FOREWORD

The information in this document has been carefully checked and is believed to be accurate; however, no responsibility is assumed for inaccuracies. SYSTRAN reserves the right to make changes without notice.

SYSTRAN makes no warranty of any kind with regard to this printed material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

©Copyright 1997 SYSTRAN Corporation. All rights reserved.

Revised: July 14, 1997

SYSTRAN Corporation
4126 Linden Avenue
Dayton, OH 45432-3068 USA
(937) 252-5601

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

TABLE OF CONTENTS

1.0 INTRODUCTION 1-1
1.1 Overview 1-1
1.1.1 Enhanced Intelligence 1-1
1.2 DTI-112X On-board Dual Port RAM 1-1
1.3 Board Features 1-2
1.3.1 Expanded Bus Controller Options 1-2
1.3.2 Monitor Terminal Function 1-2
1.3.2 1-2
1.3.3 Real Time Remote Terminal Emulation 1-2
1.3.4 Interrupt Methodology 1-2
1.3.5 Built-In Test (BIT) 1-2
1.3.5 1-2
1.3.6 Comprehensive Software 1-2
1.4 Board Specifications 1-3

2.0 GETTING STARTED 2-1
2.1 Overview 2-1
2.2 Unpacking and Examining the DTI-112X Board 2-1
2.3 Setting the Hardware Switches 2-1
2.3.1 Base Memory Address 2-2
2.3.2 Input/ Output Register Address 2-3
2.3.3 Memory Partition 2-4
2.3.4 Setting Standard and High Priority Interrupt Levels 2-5
2.4 Connecting the 112X to the Bus 2-6
2.5 Installing the DTI-112X Board 2-7
2.6 Locating the J1 And J2 Connectors 2-7

3.0 REGISTERS 3-1
3.1 Overview 3-1
3.2 Register 00 — Control Register 3-1
3.3 Register 01 — Status Register 3-3
3.4 Register 02 — Current Command Block Register 3-4
3.5 Register 03 — Polling Compare Register 3-5
3.6 Register 04 — Bit (Built-In Test) Word Register 3-6
3.7 Register 05 — Current Command or Last Valid Command Received Register 3-6
3.8 Register 06 — Interrupt Queue Pointer Register 3-7
3.9 Register 07 — High-Priority Interrupt Enable Register 3-7
3.10 Register 08 — High Priority Interrupt Status/Reset Register 3-9
3.11 Register 09 — Standard Interrupt Enable Register 3-10
3.12 Register 10 — Remote Terminal Address Register 3-10
3.13 Register 11 — BIT Start Register 3-12
3.14 Register 12 — Reset Command Register 3-12
3.15 Register 13 — RT Timer Reset Register 3-13
3.16 Register 14 — Monitor Terminal Control Register 3-13
3.17 Register 15 — Reserved 3-14
3.18 Register 16 — Monitor Terminal Address Select RT (0-15) 3-14
3.19 Register 17 — Monitor Terminal Address Select RT (16-31) 3-15
3.20 Register 18 — Master Reset/Transceiver Built-In Test 3-15
3.21 Register 19 — Interrupt Counter Read/Clear Register 3-17

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.22 Register 20 — Autodecrement Interrupt Counter Register 3-18

4.0 INTERRUPTS 4-1
4.1 Overview 4-1
4.2 Software-Programmable Registers 4-1
4.3 Hardware-Selectable Switches 4-1
4.4 Location Of Interrupt Service Routine 4-2
4.5 Interrupt Queue 4-2
4.5.1 Interrupt Status 4-3
4.5.2 Pointer 4-3
4.5.3 Tail Pointer 4-3

5.0 BUS CONTROLLER MODE 5-1
5.1 Overview 5-1
5.2 Command Block 5-1
5.2.1 Head Pointer 5-2
5.2.2 Control Word 5-2
5.2.3 Command Word One 5-5
5.2.4 Command Word Two 5-5
5.2.5 Data Pointer 5-5
5.2.6 Status Word One 5-6
5.2.7 Status Word Two 5-6
5.2.8 Tail Pointer 5-6
5.3 Linking Command Blocks 5-6
5.4 BC Functional Operation 5-7
5.4.1 BC-to-RT Transmit 5-7
5.4.2 RT-to-BC Transmit 5-8
5.4.3 RT(B)-to-RT(A) Transmit 5-8
5.5 Interrupts in the Bus Controller Mode 5-8
5.6 Mode Commands in BC Mode 5-8

6.0 REMOTE TERMINAL MODE 6-1
6.1 Overview 6-1
6.2 Host-Programmable Registers 6-1
6.3 RT Subaddress Block 6-1
6.3.1 Control Word 6-3
6.3.2 Message Status Pointer 6-4
6.3.3 Data List Pointer 6-4
6.4 Message Status Word 6-5
6.5 RT Mode Code 6-6
6.6 Mode Code Response Block 6-8
6.6.1 Control Word 6-9
6.6.2 Message Status Pointer 6-10
6.6.3 Data List Pointer 6-10

7.0 MONITOR TERMINAL MODE 7-1
7.1 Overview 7-1
7.2 Command Block 7-1
7.2.1 Control/Status Word 7-3
7.2.2 Command Word One 7-4
7.2.3 Command Word Two 7-4
7.2.4 Data Pointer 7-4
7.2.5 Status Word One 7-4
7.2.6 Status Word Two 7-4
7.2.7 Tail Pointer 7-4
7.3 Monitor Error Detection 7-4

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

7.4 Data Buffers 7-4
7.5 Monitor Terminal Registers 7-5
7.6 Data Collection 7-5
7.7 Post-Run Processing 7-6
7.8 Monitor Terminal Operational Sequence with Interrupts 7-6

8.0 BOARD PROGRAMMING 8-1
8.1 Overview 8-1
8.2 Accessing the On-board Memory and Registers 8-1
8.2.1 Memory Access 8-1
8.2.2 Creating the Interrupt Queue 8-1
8.3 Bus Controller Mode 8-3
8.3.1 Creating a Buslist 8-3
8.3.2 Initializing Data to be Transacted 8-4
8.3.3 Loading Bus Controller Registers 8-5
8.3.4 Post-Run Processing 8-5
8.4 Remote Terminal Mode 8-6
8.4.1 Subaddress/Mode Code Response Blocks 8-6
8.4.2 Initializing Data to be Transacted 8-12
8.4.3 Loading RT Registers 8-12
8.4.4 Post-Run Processing 8-12
8.5 Bus Monitor Mode 8-14
8.5.1 Initializing the Monitor Command Blocks 8-14
8.5.2 Initializing Data Buffers 8-15
8.5.3 Opening a Disk File 8-15
8.5.4 Loading Bus Monitor Registers 8-15
8.5.5 Saving Data to Disk 8-16
8.5.6 Post-Run Processing 8-16
8.6 Algorithms 8-18
8.6.1 Bus Controller Example 1 8-18
8.6.2 Bus Controller Example 2 8-19
8.6.3 Remote Terminal Example 1 8-23
8.6.4 Remote Terminal Example 2 8-26
8.6.5 Remote Terminal Example 3 8-29
8.6.6 Bus Monitor Example 1 8-32
8.6.7 Bus Monitor Example 2 8-34

9.0 DIAGNOSTICS 9-1
9.1 Overview 9-1
9.2 Tests 9-1
9.2.1 Test 1 — RAM Address Test 9-1
9.2.2 Test 2 — Walking 0's Test 9-2
9.2.3 Test 3 — Walking 1's Test 9-2
9.2.4 Test 4 — All Bits Test 9-2
9.2.5 Test 5 — Value AAAA Test 9-2
9.2.6 Test 6 — Value 5555 Test 9-2
9.2.7 Test 7 — Value 0000 Test 9-2
9.2.8 Test 8 — Register Test 9-2
9.2.9 Test 9 — Interrupt Test 9-3
9.2.10 Test 10 — Transceiver Built-In Test 9-3
9.3 Using The Diagnostics Software 9-3
9.3.1 Utility Commands 9-4
9.3.2 Test Setup And Execution Commands 9-6
9.3.3 Memory/Register Access Commands 9-9
9.4 Operator Messages 9-13

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

9.4.1 Parameter Messages 9-13
9.4.2 Status Messages 9-14
9.4.3 Test Messages 9-14
9.5 Error Messages 9-15
9.5.1 Memory Test 9-15
9.5.2 Register Test 9-15
9.5.3 Interrupt Test 9-16
9.5.4 Transceiver Built-In Test 9-16
9.6 Commands 9-16

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

FIGURES

Figure 2-1 The DTI-112X Board 2-2
Figure 2-2 Base Memory Address Factory Setting 0D000:0000 2-2
Figure 2-3 Alternative Base Address Setting 0E000:0000 2-3
Figure 2-4 Input/Output Address 0300 *hex* 2-3
Figure 2-5 Input/Output Address 0140 *hex* 2-3
Figure 2-6 Both Partitions Enabled 2-4
Figure 2-7 Partition 1 Enabled and Partition 2 Disabled 2-4
Figure 2-8 Partition 1 Disabled and Partition 2 Enabled 2-5
Figure 2-9 Standard Priority Interrupt Level 2-5
Figure 2-10 High Priority Interrupt Level 2-6
Figure 2-11 Direct Coupling 2-6
Figure 2-12 Transformer Coupling 2-7
Figure 3-1 Control Register 3-1
Figure 3-2 Status Register 3-3
Figure 3-3 Current Command Block Register 3-4
Figure 3-4 Polling Compare Register 3-5
Figure 3-5 Bit Word Register 3-6
Figure 3-6 Current Command or Last Valid Command Received Register 3-7
Figure 3-7 Interrupt Queue Pointer Register 3-7
Figure 3-8 High-Priority Interrupt Enable Register 3-8
Figure 3-9 High-Priority Interrupt Status/Reset Register 3-9
Figure 3-10 Standard Interrupt Enable Register 3-10
Figure 3-11 Remote Terminal Address Register 3-11
Figure 3-12 BIT Start Register 3-12
Figure 3-13 Reset Command Register 3-12
Figure 3-14 RT Timer Reset Register 3-13
Figure 3-15 Monitor Terminal Control Register 3-13
Figure 3-16 Monitor-Selected Remote Terminal Addresses 0-15 3-14
Figure 3-17 Monitor Selected Remote Terminal Addresses 31-16 3-15
Figure 3-18 Master Reset/Transceiver Built-In Test Register 3-16
Figure 3-19 Interrupt Counter Read/Clear Register 3-17
Figure 3-20 Autodecrement Interrupt Counter Register 3-18
Figure 4-1 Interrupt Process 4-2
Figure 4-2 Standard Priority Interrupt Queue 4-3
Figure 5-1 Bus Controller Structure 5-2
Figure 5-2 Control Word Register 5-3
Figure 5-3 BC Timing Delays 5-4
Figure 5-4 Contiguous Data Storage 5-6
Figure 5-5 Command Block Linking 5-7
Figure 6-1 RT-Subaddress Blocks 6-2
Figure 6-2 Remote Terminal Structure 6-2
Figure 6-3 RT Subaddress Control Word 6-3
Figure 6-4 Message Status Word Bits 6-5
Figure 6-5 Mode Code Control Word 6-9
Figure 7-1 Monitor Terminal Command Block Pointers 7-2
Figure 7-2 Monitor Terminal Structure 7-2
Figure 7-3 Monitor Terminal Control/Status Word 7-3
Figure 7-4 MT Command Blocks and Data Buffers 7-5
Figure 8-1 The Interrupt Queue 8-2
Figure 8-2 Bus Controller Structure 8-3
Figure 8-3 Remote Terminal Structure 8-6

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

TABLES

Table 1-1 DTI-112X Board Specifications 1-3
Table 2-1 112x J1 and J2 I/O Connectors 2-8
Table 3-1 Register 00 — Control Register 3-1
Table 3-2 Status Register 3-3
Table 3-3 Polling Compare Register 3-5
Table 3-4 Bit Word Register 3-6
Table 3-5 High Priority Interrupt Enable Register 3-8
Table 3-6 High-Priority Interrupt Status/Reset Register 3-9
Table 3-7 Standard Interrupt Enable Register 3-10
Table 3-8 Remote Terminal Address Register 3-11
Table 3-9 Monitor Terminal Control Register 3-14
Table 3-10 Master Reset Transceiver Built-In Test Register 3-16
Table 3-11 Interrupt Counter Read/Clear Register 3-17
Table 3-12 Autodecrement Interrupt Counter Register 3-18
Table 5-1 Control Word Register 5-3
Table 5-2 Causes of Interrupts 5-8
Table 6-1 RT Subaddress Control Word 6-3
Table 6-2 Message Status Word Bits 6-6
Table 6-3 Mode Codes 6-7
Table 6-4 Mode Code Control Word 6-9
Table 7-1 Monitor Terminal Control/Status Word 7-3
Table 8-1 Loading the Bus Controller Registers 8-5
Table 8-2 Function of Response Blocks 8-6
Table 8-3 Load Bus Monitor Registers 8-16
Table 9-1 Ranges for BASE Command Parameters 9-5
Table 9-2 Test Names and Numbers 9-8
Table 9-3 Memory Enable/Disable 9-9
Table 9-4 Definitions of Parameter Messages 9-13
Table 9-5 Definitions of Status Messages 9-14
Table 9-6 Definitions of Test Messages 9-14

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

1.0 INTRODUCTION

1.1 Overview

The DTI-112X is an intelligent interface between a PC-AT host and a MIL-STD-1553B Data Bus. Designed for high performance environments with multiple host applications, the 112X enables the PC-AT host to operate in real time as a MIL-STD-1553B bus controller, remote terminal or monitor terminal.

The PC-AT host processor and 112X communicate via a flexible, programmable interrupt structure, internal registers and a definable Dual Port RAM. A wide range of software-programmable and of hardware-selectable features and capabilities provide flexibility and easy use for complete MIL-STD-1553B system control, emulation analysis, and meaningful data reporting.

1.1.1 Enhanced Intelligence

The 112X's high performance architecture permits the following distributed processing solutions: bus controller-to-remote terminal, remote terminal-to-bus controller and remote terminal-to-remote terminal instructions, mode code, Broadcast Mode, and Broadcast. These solutions can be incorporated without host intervention, critical in applications that require a high degree of host processing. The following solutions can be incorporated without host intervention:

- Interrupt lists
- Multiple message processing
- Automatic polling
- Variable memory space allocation
- Selectable data storage
- Indexing

The architecture is also effective when multiple 112Xs are controlled by the same PC-AT host. This situation encompasses environments that require the following:

- Multiple MIL-STD-1553B data buses
- Multiple elements (BCs, RTs, MTs, etc.)

1.2 DTI-112X On-board Dual Port RAM

To maximize processing, the 112X contains 32K words of Dual Port RAM. The RAM size may be altered to enhance data handling. The RAM configuration may be altered to meet specific user requirements.

To meet a wide range of systems requirements, the 112X allows you to define not only the size of the Dual Port RAM (16K and 32K), but also the starting address of the memory block. The 112X RAM can be enabled and disabled, allowing use of the same starting address for many boards. The user can optionally increase the number of boards, according to available board space and power.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

The Dual Port RAM is accessible directly from the bus or the on-board MIL-STD-1553B. Since byte READs and WRITEs produce unpredictable results, access all Dual Port RAM in full-word transfers

1.3 Board Features

1.3.1 Expanded Bus Controller Options

The 112X initiates and controls the execution of multiple lists of bus instructions. The 112X also emulates the instruction execution and the specific timing relationship on the MIL-STD-1553B Bus without host intervention.

1.3.2 Monitor Terminal Function

Serving as a monitor terminal, the 112X monitors a specific remote terminal, a group of remote terminals or all remote terminals.

1.3.3 Real Time Remote Terminal Emulation

The 112X responds to bus instructions in real time, easily handling the traffic of a 100 percent loaded bus. The 112X is capable of transmitting, or receiving and storing, 1000 bus messages of 32 data words each.

1.3.4 Interrupt Methodology

A sophisticated two-tier interrupt methodology is permits the customization of interrupt handling to meet the demands of specific customer applications.

1.3.5 Built-In Test (BIT)

The 112X ensures the functionality of each board with a self-test. Self-testing is performed by passing data through the transceiver or data loops. With data loops, data need not go from the 112X onto the MIL-STD-1553B.

1.3.6 Comprehensive Software

The application provides ergonomically designed, menu-driven software for the 112X. A help screen-oriented Diagnostics and Command Library provides compatible tools for the developer.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

1.4 Board Specifications

The board specifications for the 112X are shown in Table 1-1.

Table 1-1 DTI-112X Board Specifications

Specification	Value
Memory	32 K Words of Dual Port RAM
Word Size	16 bits
Communication	MIL-STD-1553B protocol Dual redundant bus
Interface to bus	Transformer or direct coupling
Voltage	+5.0V ± 5%, +12.0V ±5%, -12.0V ±5%
Current Drain	3.5 amperes (max.)
Temperature Range: Operating Storage	 0 to 70 degrees Celsius -55 to 125 degrees Celsius
Relative Humidity	10% to 90%, non-condensing
DTI-1120/1121 Board Size	332.74 mm x 111.76 mm

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.0 GETTING STARTED

2.1 Overview

There are five steps involved in setting up the DTI-112X board :

- Unpacking and examining the board
- Setting the hardware switches
- Connecting the board to the bus
- Installing the board
- Locating the J1 and J2 connectors

2.2 Unpacking and Examining the DTI-112X Board

Remove the board from the packing box, setting packing foam aside for future use.



NOTE: The 112X is shipped in a protective bag. Do not remove the board from the anti-static bag until the board is grounded.

Once grounded, remove the hardware module from the protective bag. Place the board on top of the protective bag or on an electrostatically-controlled work surface.

If the board appears to be damaged, contact SYSTRAN Corporation immediately at **(937) 252-5601** between 8 AM - 5 PM EST.

2.3 Setting the Hardware Switches



NOTE: Extended Memory, Input/Output Memory and On-board Dual Port Ram addresses are contained in hardware-selectable switches that must be set before inserting the DTI-112X in the PC-AT host.

The 112X board houses five switch banks that are numbered consecutively and located across the top of the board (Figure 2-1). A switch bank is initiated by setting each of the switches, eight (on Switch Banks 1 through 3) or ten (on Switch Banks 4 and 5) to the OPEN or CLOSED position.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

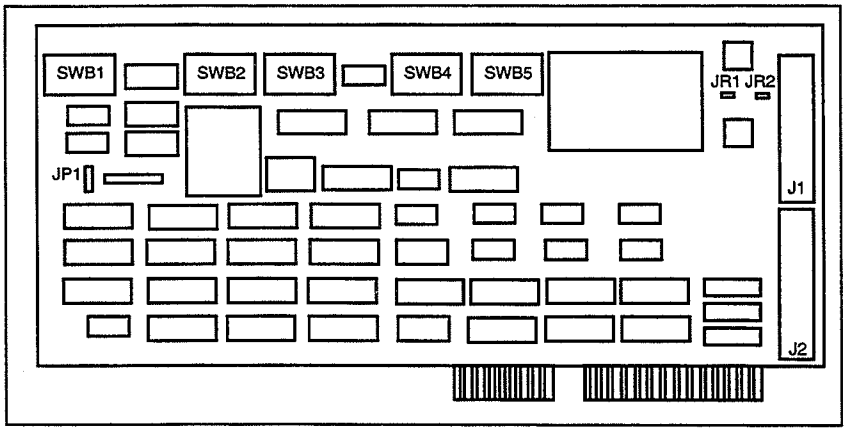


Figure 2-1 The DTI-112X Board

NOTE: Labels MSB and LSB denote the most significant and least significant bits, respectively.

2.3.1 Base Memory Address

The 112X requires a full page of unused memory. Switch Bank 1 sets the Base Memory Address. Figure 2-2 shows the factory-set Base Memory Address 0D000:0000.

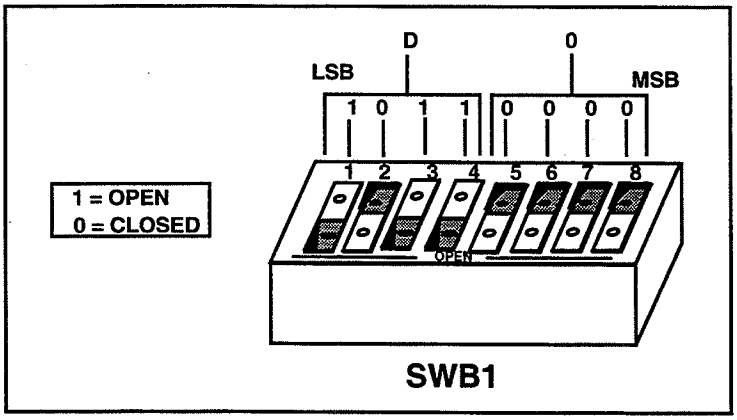


Figure 2-2 Base Memory Address Factory Setting 0D000:0000

Figure 2-3 provides an alternative example for Base Memory addressing, using Address 0E000:0000.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

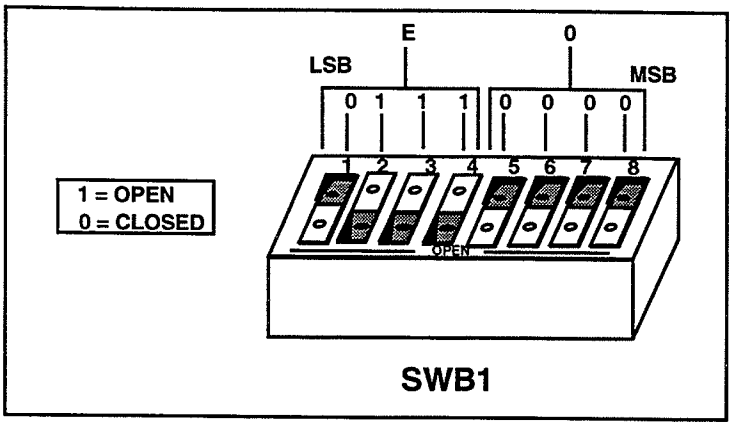


Figure 2-3 Alternative Base Address Setting 0E000:0000

2.3.2 Input/ Output Register Address

Switch Banks 2 and 3 set the Input/Output Register addresses and store the reserved host-accessible Input/Output Register address. The 112X requires a 40 *hex* block of I/O address. When setting the switches, ensure the board is based on a 40 *hex* boundary. Figure 2-4 shows the I/O address factory setting of 0300 *hex*.

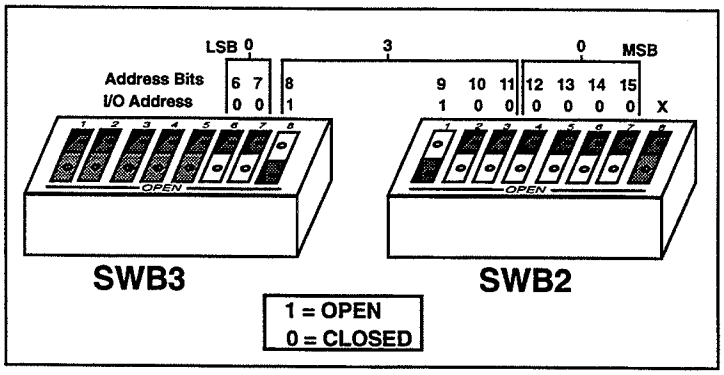


Figure 2-4 Input/Output Address 0300 *hex*

Figure 2-5 provides an alternative example for I/O addressing, using Address 0140 *hex*.

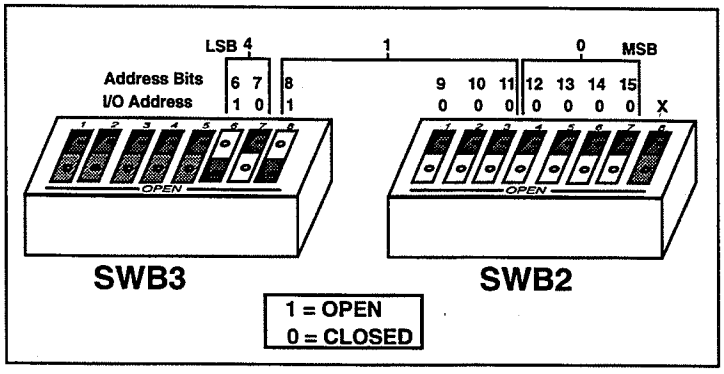


Figure 2-5 Input/Output Address 0140 *hex*

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

2.3.3 Memory Partition

Switches 4 and 5 of Switch Bank 3 set the Memory Partition. Switch 4 sets the first half of the Memory Partition (X000:0000h to X000:3FFFh). Switch 5 sets the second half of the Memory Partition (X000:4000h to X000:7FFF).

The amount of Dual Port RAM available for data handling can be altered to 32K or 16K words by changing the switch settings on Switch Bank 3. Two memory partitions can be created.

To enable both partitions, set Switches 4 and 5 to the CLOSED position as shown in Figure 2-6.

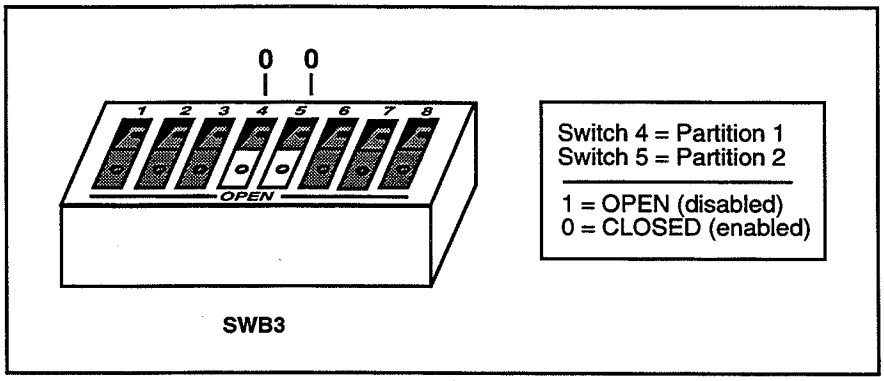


Figure 2-6 Both Partitions Enabled

To enable Partition 1, set Switch 4 CLOSED and Switch 5 OPEN (Figure 2-7).

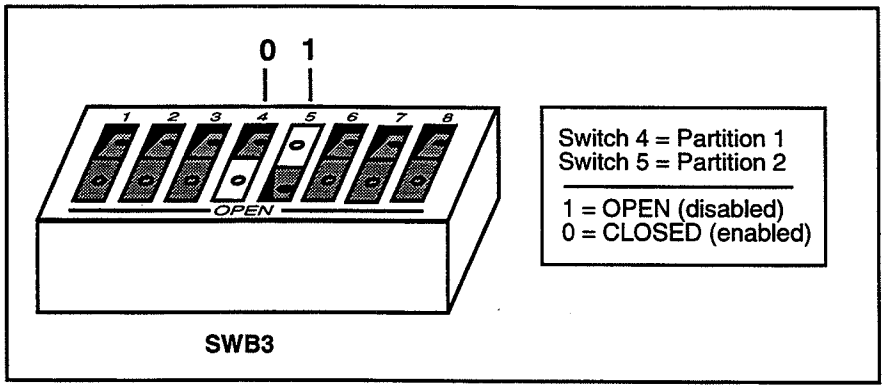


Figure 2-7 Partition 1 Enabled and Partition 2 Disabled

To enable Partition 2, set Switch 4 OPEN and Switch 5 CLOSED (Figure 2-8).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

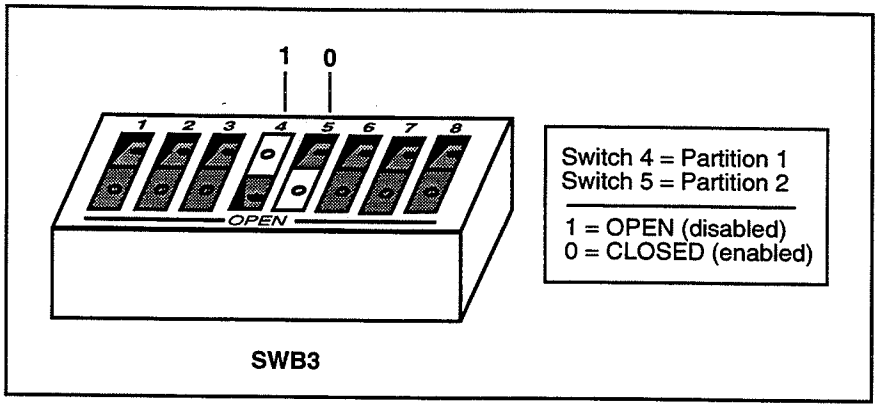


Figure 2-8 Partition 1 Disabled and Partition 2 Enabled

NOTE: The three least significant bits of the lower digit (Switch Bank 3) are unused and should be read as zeros.

2.3.4 Setting Standard and High Priority Interrupt Levels

Refer to Chapter 4, Interrupts, for more detailed information on interrupts.

STANDARD PRIORITY INTERRUPT LEVEL

Switch Bank 4 sets the system level (3-7, 10-12, 15) of the Standard Priority Interrupt Level. Select an interrupt level by depressing its corresponding switch to the CLOSED position. Figure 2-9 shows Interrupt Level 12 being selected.

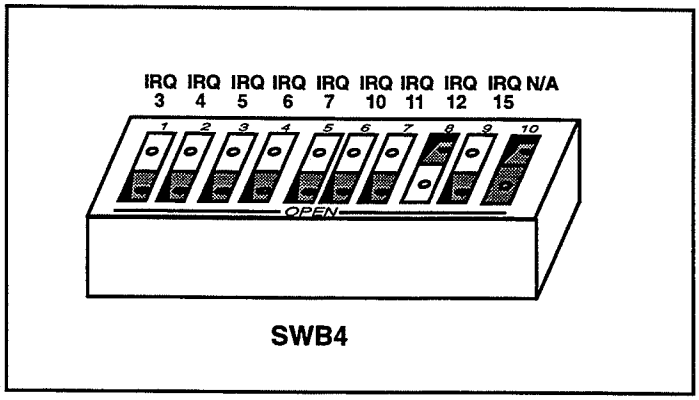


Figure 2-9 Standard Priority Interrupt Level

NOTE: Do not select more than one interrupt level at a time.

HIGH PRIORITY INTERRUPT LEVEL

Switch Bank 5 sets the System Level (3-7, 10-12, 15) of the High Priority Interrupt Level. To pass a selected interrupt level, depress the interrupt level's corresponding switch to the CLOSED position. Figure 2-10 shows Interrupt Level 11 being selected.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

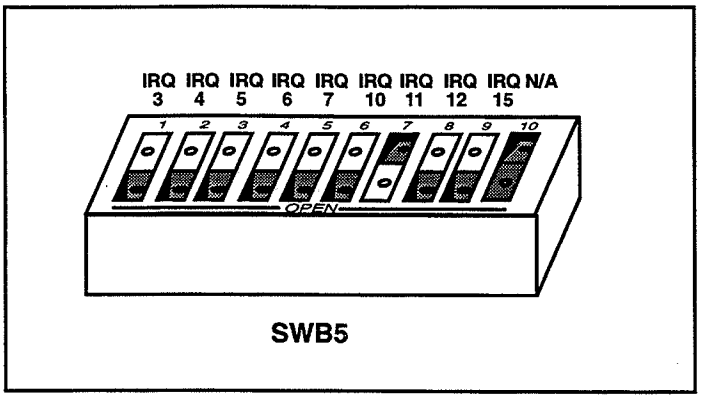


Figure 2-10 High Priority Interrupt Level

NOTE: Do not select more than one interrupt level at a time.

2.4 Connecting the 112X to the Bus

The 112X is connected to the MIL-STD-1553B bus using either a Direct or Transformer Coupled MIL-STD-1553B I/F Cable assembly (available through DTI). Both coupling types are available on I/O Connector J1. To change the type of coupling, change the cable configuration.

Use Direct Coupling if the distance between the 112X and the MIL-STD-1553B bus equals 12 inches or less (Figure 2-11).

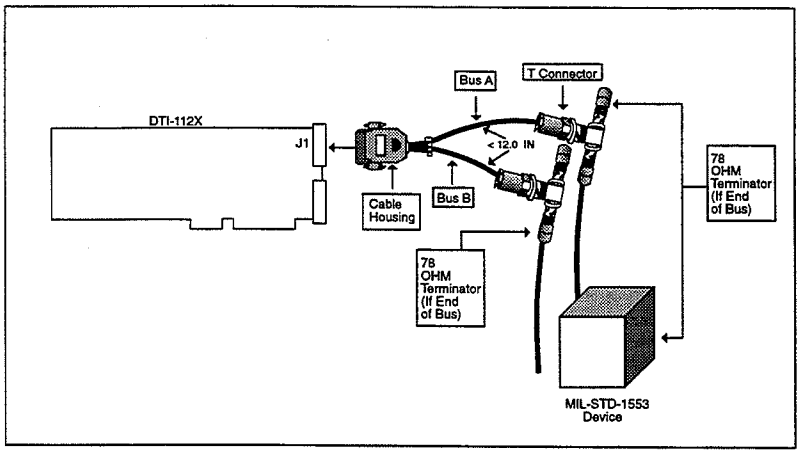


Figure 2-11 Direct Coupling

NOTE: Buses A and B must contain two 78 OHM terminators if not connected to a device or MIL-STD-1553B bus.

Use Transformer Coupling if the distance between the 112X and the MIL-STD-1553B bus exceeds 12 inches but no more than 20 feet (Figure 2-12).

Artisan Technology Group | Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

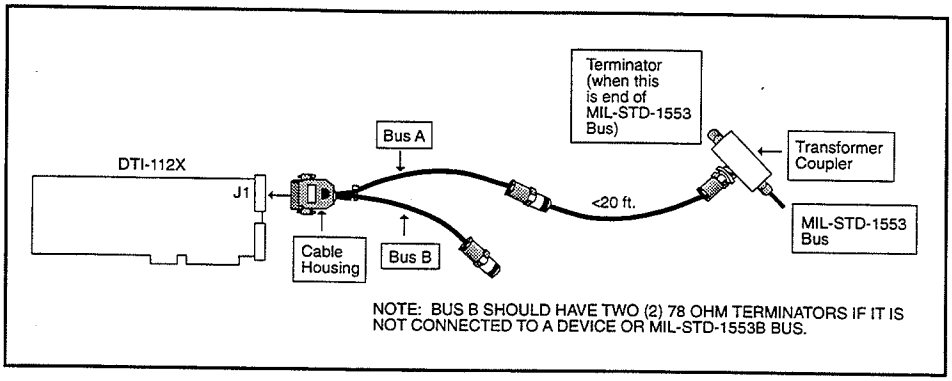


Figure 2-12 Transformer Coupling

2.5 Installing the DTI-112X Board

Once the switch banks are set, the 112X can be installed in the host.

CAUTION: Ensure power to the computer is off before installing the board.

Position the board over an available AT slot

NOTE: An AT slot must be used in order to ensure the AT connector fits securely in the slot.

Push down firmly on top of the board until it locks into place. Secure front panel with a screw.

2.6 Locating the J1 And J2 Connectors

Two 15-pin connectors, J1 and J2, are located on the right side of the 112X bracket (Table 2-1). The top connector, J1, allows communication to the MIL-STD-1553B bus in both Direct and Transformer Coupling. The J2 Connector synchronizes multiple 112X Boards (for inter-active timing between the 112X and host) .

The J2 Port Signals follow :

- **Pins 1 and 15** Ground
  These pins can be used to ground connections.
- **Pin 2** Sub-System Fail (Input)
  When set, this bit sets the SUBSYSTEM FAIL Bit in Register 1. An interrupt can then be enabled, notifying the host. Pin 2 detects test hardware failure (such as a MIL-STD-1553B test equipment failure).
- **Pin 3** Command Strobe (Output)
  The 112X activates this pin after receiving a valid MIL- STD-1553B command and deactivates the pin after servicing the command.
- **Pin 4** External Override (Input)
  This pin is used for INPUT from the Remote Terminal Command Strobe. It can also be used in a multi-redundant application. Upon

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

receiving the Pin 3 signal, all current activity is aborted. The pin can be connected to the RT Command Strobe OUTPUT of an adjacent 112X in use.

- **Pins 5-14** Reserved

Table 2-1 112x J1 and J2 I/O Connectors

	J1 DB-15S (insert in top of bracket)		J2 DB-15S (insert in bottom of bracket)	
	Pin	**Signal**	**Pin**	**Signal**
Direct Coupled	1	BUS A DATA	1	D.GND
	2	BUS A DATA*	2	SYSF
	3	RESERVED	3	COMSTB*
	4	RESERVED	4	EXTOVR*
	5	RESERVED	5	RESERVED
Direct Coupled	6	BUS B DATA	6	RESERVED
	7	BUS B DATA*	7	RESERVED
	8	CHASSIS GND	8	RESERVED
	9	RESERVED	9	RESERVED
	10	RESERVED	10	RESERVED
Transformer Coupled	11	BUS A DATA (L)	11	RESERVED
	12	BUS A DATA*(L)	12	RESERVED
	13	RESERVED	13	RESERVED
Transformer Coupled	14	BUS B DATA (L)	14	RESERVED
	15	BUS B DATA*	15	D.GND

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.0 REGISTERS

3.1 Overview

The 112X houses 21 16-bit registers that define board characteristics and operational modes.

3.2 Register 00 — Control Register

Once the Control Register is written, interface between the 112X and the 1553 Bus can be initiated (Figure 3-1).

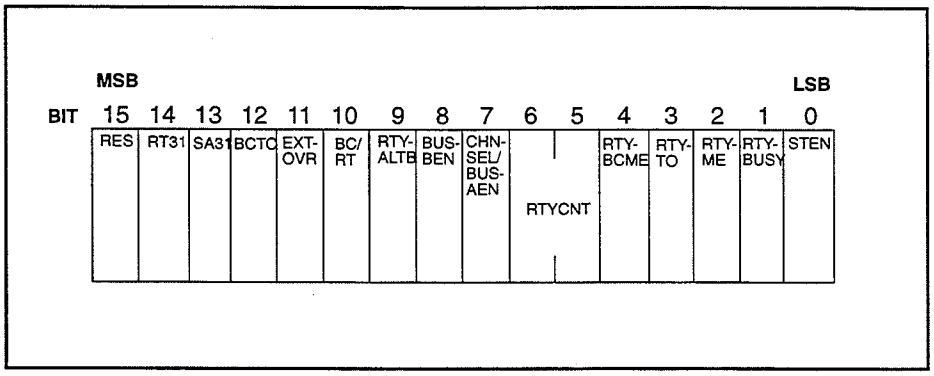


Figure 3-1 Control Register

Each bit in the Control Register is described in Table 3-1.

Table 3-1 Register 00 — Control Register

Bit	Function	Description
0	Start Enable (STEN)	In BC Mode, Bit 0 starts/restarts command block execution. In RT (or MT) Mode, Bit 0 enables the 112X to receive a valid command. RT operation does not begin until a valid command is received. When using STEN: • Restart the 112X after each Master Reset or programmed reset. • Verify operation by reading Bit 0 of the Status Register 01 (since Bit 0 of Register 0 is unreadable).
1	Retry On Busy (RTYBSY)	In BC Mode, Bit 1 enables automatic retry on a received busy bit in an RT status word response.
2	Retry On Message Error (RTYME)	In BC Mode, Bit 2 enables an automatic retry when the message error bit is set in the RT's status word response.
3	Retry On Time-Out (RTYTO)	In BC Mode, Bit 3 enables an automatic retry on a response time-out condition.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 3-1 Register 00 — Control Register

Bit	Function	Description
4	Retry on Bus Controller Message Error (RTYBCME)	In BC Mode, Bit 4 enables automatic retries on an error the bus controller detects.
5-6	Retry Count (RTYCNT)	In BC Mode, Bits 5 and 6 program the number of retries (up to four) to attempt: 00 = one retry 01 = two retries 10 = three retries 11 = four retries
7	Channel A Enable (CHNSEL /BUSAEN)	In RT (or MT) Mode, setting Bit 7 enables Bus A. In BC Mode, Bit 7 selects the communication bus: 1 enables Bus A 0 enables Bus B
8	Channel B Enable (BUSBEN)	In RT (or MT) Mode, setting Bit 8 enables Channel B . Bit 8 is undefined in BC Mode.
9	Retry on Alternate Bus (RTYALTB)	In BC Mode, Bit 9 enables an automatic retry to operate on the alternate bus. For example, if operating on Bus A, with two automatic retries, the retries occur on Bus B. Bit 9 is undefined in RT (or MT) Mode.
10	BC/RT* Select (BC/RT)	Bit 10 selects the 112X operation mode: 1 = BC Mode 0 = RT (or MT) Mode Bit 10 must be reset to 0, when shifting from BC to RT (or MT) Mode.
11	Enable External Override (EXTOVR)	Bit 11 enables external override (used when operating multiple boards). Set Bit 11 and ground the external port Pin 3.
12	Bus Controller Time-Out (BCTO)	When the 112X is a BC and BCTO = 0, the 112X allows an RT up to 16µs to respond with a Status Word before declaring a bus time-out. If BCTO = 1, the 112X allows an RT up to 32µs to respond with a status word before declaring a time-out.
13	Subaddress 31 (SA31)	When SA31 = 0, the 112X recognizes a command word with either Subaddress 0 or 31 as a valid mode code. When SA31 = 1, the 112X only recognizes a command word with Subaddress 0 as a valid mode code.
14	RT Address 31 (RT31)	When RT31 = 0, the 112X recognizes RT Address 31 as a Broadcast Command. When RT31 = 1, the 112X recognizes RT Address 31 as a normal terminal address.
15	Reserved	

NOTE: Bits 1-6 are undefined in RT (or MT) Mode.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.3 Register 01 — Status Register

The read-only Status Register (Figure 3-2) indicates the 112X's current status.

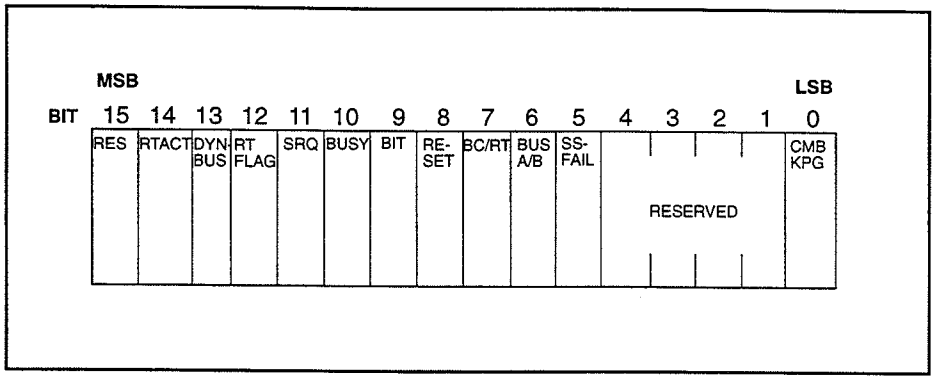


Figure 3-2 Status Register

Each bit in the Status Register is described in Table 3-2.

Table 3-2 Status Register

Bit	Function	Description
0	(CMBKPG)	If Bit 0 is set, either Command Block execution is in progress (BC Mode) or RT (MT) Mode is operating.
1-4	Reserved	
5	Subsystem Fail Indicator (SSFAIL)	If set, Bit 5 indicates it has received a subsystem fail signal from the host subsystem in response to SSYSF input.
6	Channel A/B* (BUSA/B)	Bit 6 indicates the bus in use or the last bus used: Bus A = 1 (On) Bus B = 0 (Off)
7	BC/(RT* or MT*) Mode (BC/RT)	Bit 7 indicates the current mode of operation: BC = 1 (On) RT (or MT) = 0 (Off)
8	Reset in Progress (RESET)	Bit 8 indicates that either a write Register 12 has just occurred or the 112X has just received a Reset Remote Terminal (#0100) Mode Code. Bit 8 remains set less than one micro-second.
9	Built-in Test (BIT)	Bit 9 indicates the Built-In Test (BIT) is in progress.
10	Busy Bit (BUSY)	In RT (or MT) Mode, Bit 10 indicates the Busy Bit is set in RT status word, reflecting the result of writing to Register 10, Bits 9 or 14.

NOTE: A reset operation must be performed when changing between BC and RT Modes.

11	Service Request	In RT (or MT) Mode, Bit 11 indicates the Service Request Bit is set in the RT status word, reflecting the

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 3-2 Status Register

Bit	Function	Description
	(SRQ)	result of writing to Register 10, Bit 10.
12	Terminal Flag (RT FLAG)	In RT (or MT) Mode, Bit 12 indicates the Terminal Flag Bit is set in the RT status word, reflecting the result of writing to Register 10, Bit 11.
13	Dynamic Bus Control Acceptance (DYNBUS)	In RT (or MT) Mode , Bit 13 reflects the status of the Dynamic Bus Control Acceptance Bit in the RT status word.
14	Remote Terminal (or Monitor) Active (RTACT)	In RT (or MT) Mode, Bit 14 indicates that the 112X, in the RT (or Monitor) Mode, is presently servicing a command.
15	Reserved	

NOTE: Bits 10-14 are undefined in BC (or MT) Mode.

3.4 Register 02 — Current Command Block Register

The Current Command Block Register (Figure 3-3) is the register for the Buslist Instruction Address, the Subaddress Response Block Address, and the Next Buslist Instruction

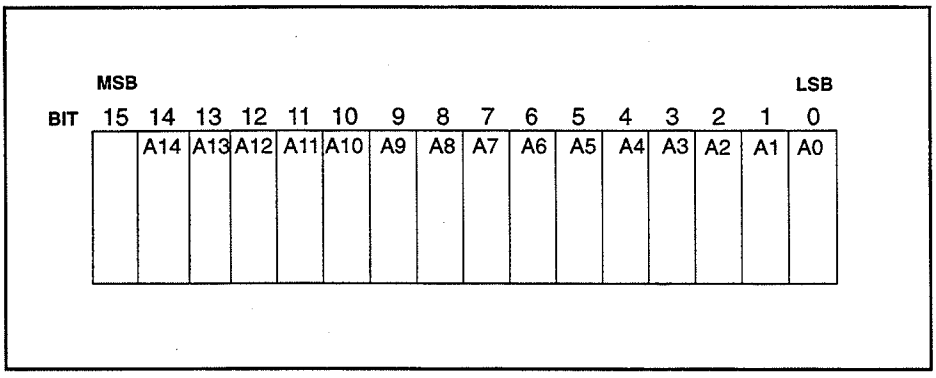


Figure 3-3 Current Command Block Register

Bits 0-15 are used in the following way when in the mode described:

BC MODE

The Command Block Register contains the head pointer address of the command block being executed. Accessing a new command block updates the address.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

RT MODE

The host CPU initializes the Command Block Register to indicate the starting address of the RT Descriptor Space. The host must allocate 320 sequential locations following the starting address. For proper operation, the address must start on an I x 512 decimal boundary, where "I" is an integer multiple.

MONITOR MODE

The Command Block Register contains the control/status word address of the current monitor command block. Accessing a new command block updates the address.

3.5 Register 03 — Polling Compare Register

Figure 3-4 illustrates the Polling Compare Register and Table 3-3 describes the bits in this register.

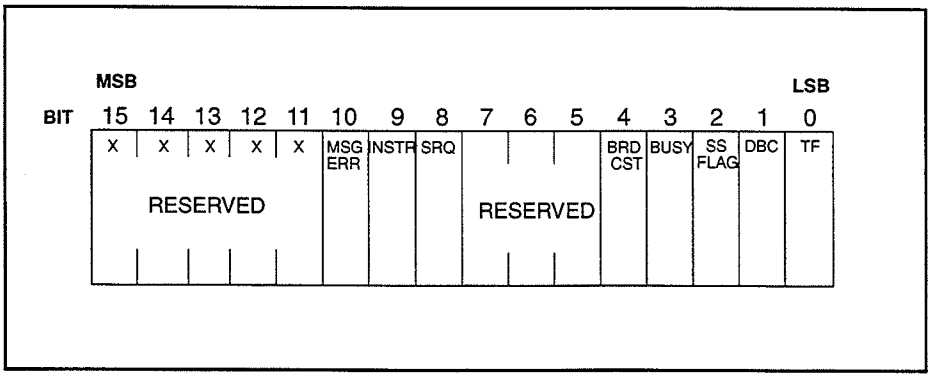


Figure 3-4 Polling Compare Register

Table 3-3 Polling Compare Register

Bit	Function	Description
0-10		In the Polling Mode, the CPU sets the Polling Compare Register to indicate the RT response word that causes the 112X to interrupt. The remote terminal response is compared to the Polling Compare Register. If any of the 11 least significant bits (Bits 0-10) of the status word are set and the corresponding bit is set in the Polling Compare Register, a Polling Comparison Interrupt is generated. The Polling Compare Register's 11 bits correspond to Bits 9 through 19 of the RT's 1553 status word response. The RT's sync, remote terminal address and parity bits are not included.
11-15	Reserved	

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.6 Register 04 — Bit (Built-In Test) Word Register

The 112X uses the Bit Word Register (Figure 3-5) when responding to the Transmit BIT Word Mode Code (#10011). A "1" written to either Bit 14 or 15 of the register induces a BIT failure.

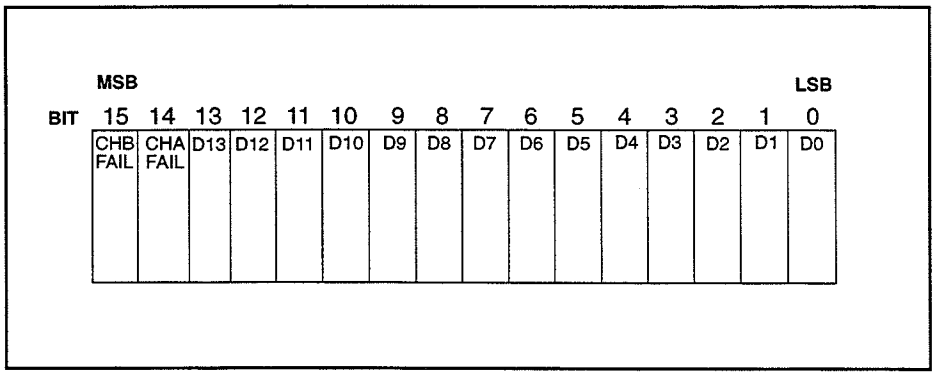


Figure 3-5 Bit Word Register

The bits in the Bit Word Register are described in Table 3-4.

Table 3-4 Bit Word Register

Bit	Function	Description
0-13	BIT Word (D0 - D13)	Bits 0 through 13 of the BIT Word are user programmable. The 112X writes to the two most significant bits of the BIT Word Register in response to either an Initiate Self-Test Mode Code (RT Mode) or a write to Register 11, Bit Start Command. If the BIT Word requires modification, it can be read out, modified and rewritten to the Bit Word Register.
14	Channel A (CHAFAIL)	Bit 14 indicates a Bus A self-test failure.
15	Channel B (CHBFAIL)	Bit 15 indicates a Bus B self-test failure.

NOTE: Writing a '1' to either Bit 14 or 15 induces a BIT failure.

3.7 Register 05 — Current Command or Last Valid Command Received Register

In RT or Monitor Mode, the Current Command Register (Figure 3-6) contains the command currently being processed. When not processing a command, the 112X stores the last command word transmitted on the 1553B Bus in the Current Command Register. The register is updated only when Bit 0 of Register 0 is set.

In the BC Mode, the Current Command Register contains the most current command sent out on the 1553B Bus.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

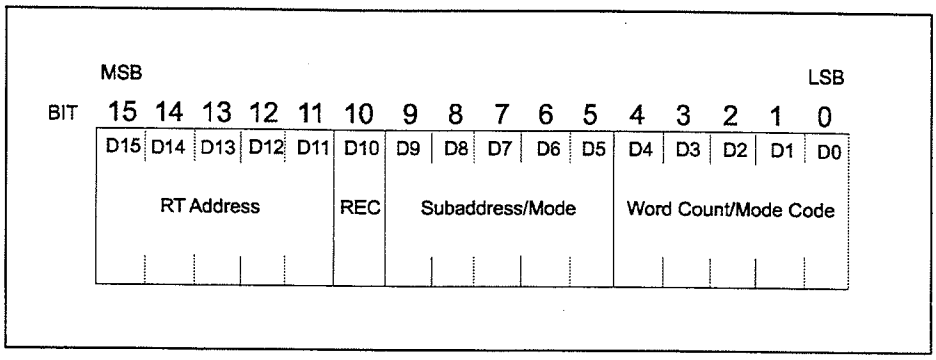


Figure 3-6 Current Command or Last Valid Command Received Register

3.8 Register 06 — Interrupt Queue Pointer Register

Initialized by the CPU, the Interrupt Queue Pointer Register (Figure 3-7) contains the starting location of the Standard Priority Interrupt Queue. The queue is updated with the address of the next queue entry when an interrupt block is added to the queue.

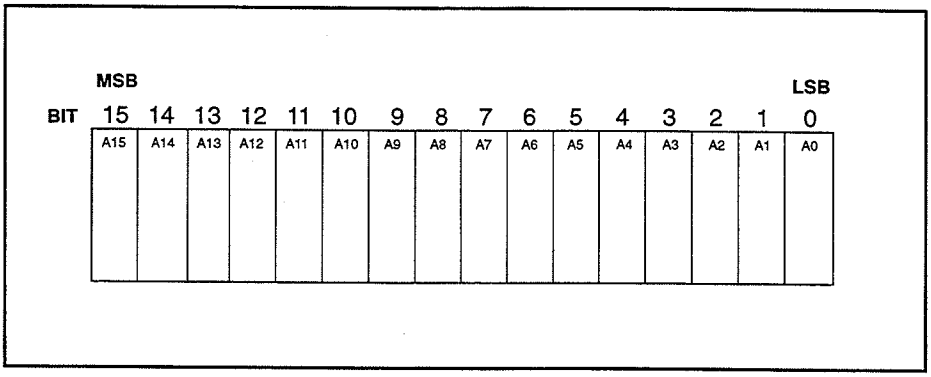


Figure 3-7 Interrupt Queue Pointer Register

3.9 Register 07 — High-Priority Interrupt Enable Register

Setting the bits in the High-Priority Interrupt Enable Register (Figure 3-8) causes a High-Priority Interrupt when the enabled event occurs.

To service the High-Priority Interrupt, read Register 8 to determine the cause of the interrupt. Then write to Register 8 to clear the appropriate bits. The 112X also provides a Standard-Priority Interrupt scheme that requires no host intervention.

When High-Priority Interrupt service is impossible in a given application, use the Standard-Priority features.

The bits in the High-Priority Interrupt Enable Register are described in Table 3-5.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

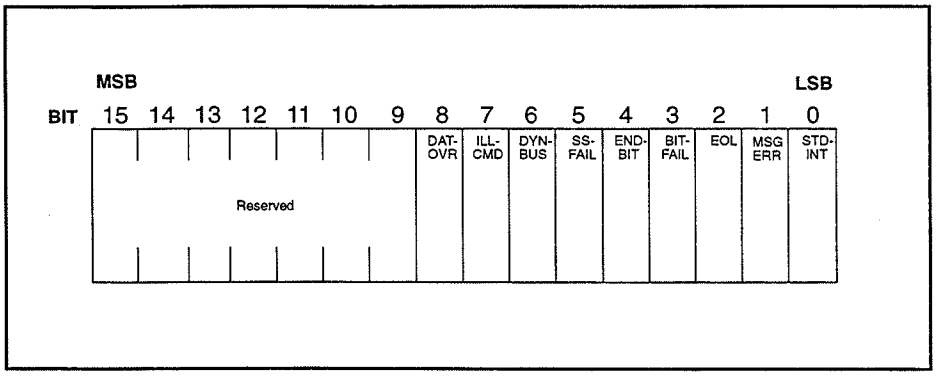


Figure 3-8 High-Priority Interrupt Enable Register

Table 3-5 High Priority Interrupt Enable Register

Bit	Function	Description
0	Standard Interrupt Enable (STDINT)	Setting Bit 0 enables all Standard Interrupts for BC or RT (or MT), without causing a High-Priority Interrupt.
1	Message Error Enable (MSGERR)	If Bit 1 is enabled, a High-Priority Interrupt is asserted at the occurrence of a message error. If a High-Priority Interrupt condition occurs as the result of an enabled message error, the device halts operation until the user clears the interrupt by writing a "1" to Bit 1 of the High-Priority Interrupt Status/Reset Register (Register 08). If the interrupt is not cleared, the 112X remains in the halted state (appearing to be locked-up) even when receiving a valid message. The Message Error Enable scheme is required to maintain the 112X's operational state and to ensure the host CPU has access to error message information during interrupt service.
2	End of Command Block List Enable (EOL)	In BC Mode, Bit 2 indicates the end of the buslist. The Bit 2 Interrupt can be superseded by other High-Priority Interrupts.
3	BIT Word Fail Enable (BITFAIL)	Bit 3 enables an interrupt indicating the 112X detects a BIT failure.
4	End of Bit Enable (ENDBIT)	Bit 4 indicates the end of the internal Built-In Test (BIT).
5	Subsystem Fail Enable (SSFAIL)	When Bit 5 is set, a High-Priority Interrupt is asserted after receiving a Subsystem Fail (SSYSF) input pin.
6	Dynamic Bus Control Mode Code Interrupt Enable (DYNBUS)	When Bit 6 is set, the 112X asserts an interrupt when the Dynamic Bus Control Mode Code is received.
7	Illogical Command Error Enable (ILLCMD)	In BC Mode, Bit 7 enables a High-Priority Interrupt, asserted when an illogical command (such as an incorrectly formatted RT-RT Command Block) occurs.
8	Data Overrun Enable (DATOVR)	When set, Bit 8 enables an interrupt when the 112X does not receive DMA grant within the time allotted for successful data transfer to memory.
9-15	Reserved	

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.10 Register 08 — High Priority Interrupt Status/Reset Register

When a High-Priority Interrupt is asserted, Register 8 (Figure 3-9) identifies the event that caused it. To clear the interrupt signal and reset the bit, write a '1' to the appropriate bit. See corresponding bit definitions of Register 7 (Figure 3-8), the High-Priority Interrupt Enable Register.

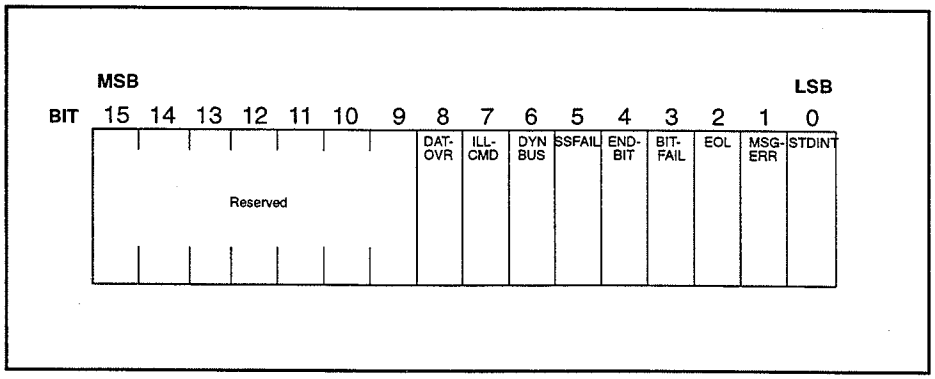


Figure 3-9 High-Priority Interrupt Status/Reset Register

The bits in the High-Priority Interrupt Status/Reset Register are described in Table 3-6.

Table 3-6 High-Priority Interrupt Status/Reset Register

Bit	Function	Description
0	Standard Interrupt (STDINT)	The 112X sets Bit 0 when any Standard Interrupt occurs, providing Bit 0 of Register 7 is enabled.
1	BC, RT or MT Message Error (MSGERR)	
2	End of Command Block (EOL)	
3	BIT Word Fail (BITFAIL)	
4	End of BIT (ENDBIT)	
5	Subsystem Fail (SSFAIL)	
6	Dynamic Bus Control Mode Code Received (DYNBUS)	
7	Illogical Command (ILLCMD)	
8	Data Overrun (DATOVR)	
9-15	Reserved	

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.11 Register 09 — Standard Interrupt Enable Register

Register 9 (Figure 3-10) enables Standard Interrupt logging for any of the enabled events described below. Standard Interrupt logging can also occur for events enabled in the BC Command Block, RT Subaddress/Mode Code Descriptor or Monitor Command Block.

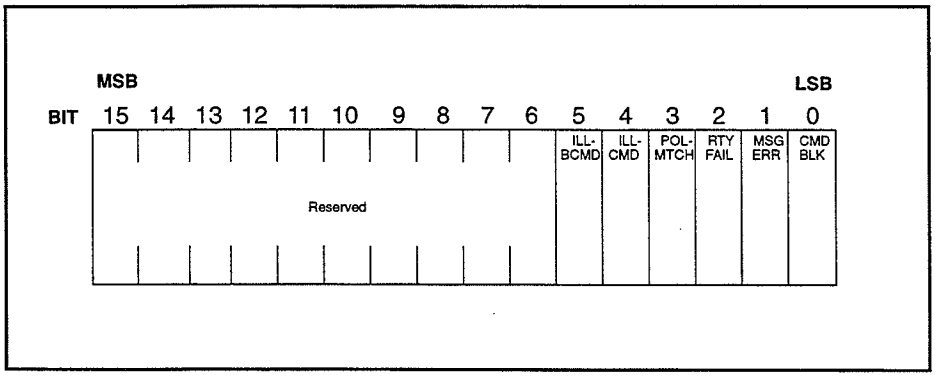


Figure 3-10 Standard Interrupt Enable Register

The bits in the Standard Interrupt Enable Register are described in Table 3-7.

Table 3-7 Standard Interrupt Enable Register

Bit	Function	Description
0	Command Block Interrupt and Continue (CMDBLK)	In BC and MT Modes, Bit 0 enables an interrupt indicating a command block, with the Interrupt and Continue Function enabled, has been executed.
1	Message Error Event (MSGERR)	In BC, RT and MT modes, Bit 1 enables a Standard Interrupt for message errors.
2	Retry Fail (RTYFAIL)	In BC Mode, Bit 2 enables an interrupt indicating all the programmed number of retries have failed.
3	Polling Comparison Match (POLMTCH)	In BC Mode, Bit 3 enables an interrupt indicating a polling event has occurred. The user must also set Bit 12 in the BC Command Block Control Word for the POLMTCH Interrupt to occur.
4	Illegal Command (ILLCMD)	In RT Mode, when Bit 4 is set, it enables an interrupt indicating that an illegal command has been received.
5	Illegal Broadcast Command (ILLBCMD)	In RT Mode, when Bit 5 is set, it enables an interrupt indicating an Illegal Broadcast Command has been received.
6-15	Reserved	

3.12 Register 10 — Remote Terminal Address Register

Register 10 (Figure 3-11) sets the RT address via software. When set, the Change Lock-Out Enable feature prevents the RT address or the 112X mode selection from changing.

The bits in the Remote Terminal Address Register are described in Table 3-8.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

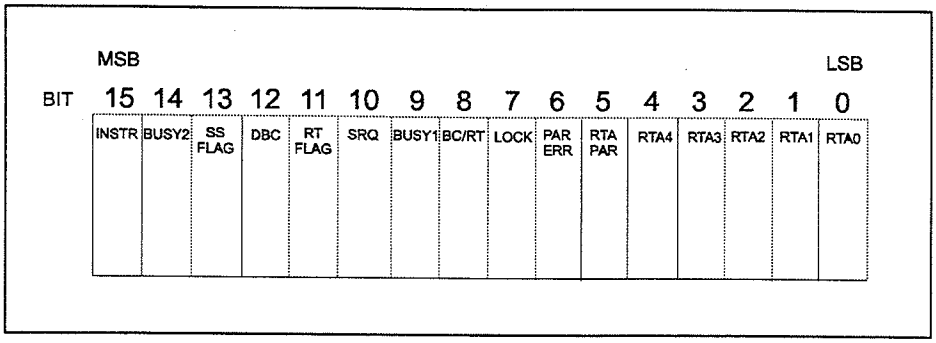


Figure 3-11 Remote Terminal Address Register

Table 3-8 Remote Terminal Address Register

Bit	Function	Description
0-4	Remote Terminal Address (RTA0-4)	The RT address reflects the RTA0-4 inputs at Master Reset. Modify the remote terminal address by writing to Bits 0-4.
5	Remote Terminal Address Parity (RTAPAR)	Bit 5 is an odd parity bit used with the RT address. Bit 5 ensures accurate recognition of the RT address.
6	Remote Terminal Address Parity Error (PARERR)	Bit 6 indicates a Remote Terminal Address Parity Error has occurred. If a parity error exists, the error is indicated after the remote terminal address is latched.
7	Change Lock-out (LOCK)	When set, Bit 7 indicates changes to the RT address or the BC/RT (or MT) Mode Select are not allowed using internal registers. Bit 7 is read-only.
8	BC/RT Mode Select (BC/RT*)	Bit 8's state indicates the internal BCRTSEL signal is set. Bit 8 is read-only.
9	Busy Mode Enable (BUSY1)	In RT Mode, Setting Bit 9 sets the RT Status Word Busy Bit and inhibits all data transfers to the subsystem.
10	Service Request (SRQ)	In RT Mode, setting Bit 10 sets the RT Status Word Service Request Bit.
11	Terminal Flag (RT FLAG)	In RT Mode, setting Bit 11 sets the RT Status Word Terminal Flag Bit. The Terminal Flag Bit in the RT status word is also internally set if the BIT fails.
12	Bit Dynamic Bus Control Acceptance (DBC)	Setting Bit 12 sets the RT Status Word Dynamic Bus Control Acceptance Bit when the 112X receives the Dynamic Bus Control Mode Code from the currently active bus controller. Host intervention is required for the 112X to take over as the active bus controller.
13	Subsystem Fail (SS Flag)	Setting Bit 13 sets the RT Status Word Subsystem Flag Bit. In the RT Mode, the Subsystem Fail is also logged into the message status word.
14	Busy (BUSY2)	Setting Bit 14 sets the RT Status Word Subsystem Flag Bit. In the RT Mode the Subsystem Fail is also logged into the message status word.
15	Instrumentation (INSTR)	Setting Bit 15 sets the RT Status Word Instrumentation Bit.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.13 Register 11 — BIT Start Register

Any write (such as, data = don't care) to Register 11's address location initiates the internal BIT routine, which lasts 100µs. Verify using the BIT-In-Progress Bit in the Status Register. A programmed reset (write to Register 12) must precede a write to Register 11 to initiate the internal BIT.

The board's self-test performs an internal wrap-around test between its Manchester encoder and its two Manchester decoders. If the 112X detects a failure on either the primary or the secondary channel, it flags the failure by setting Bit 14 of Register 4 (BIT WORD Register) for Channel A and/or Bit 15 for Channel B. In the Remote Terminal Mode, during the board self-test, the 112X ignores any commands on the 1553 bus until the self-test is completed.

Figure 3-12 illustrates the BIT Start Register.

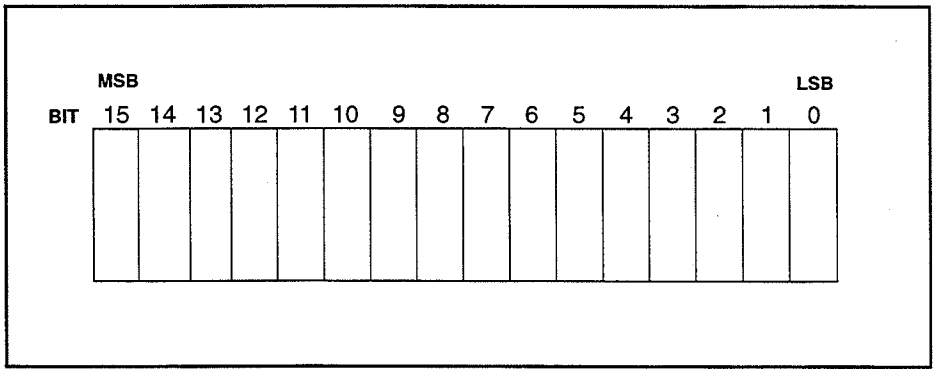


Figure 3-12 BIT Start Register

3.14 Register 12 — Reset Command Register

Any write (such as, data = don't care) to Register 12's address location initiates a reset sequence of the encoder/decoder and protocol sections of the 112X that lasts less than one microsecond. The reset is identical to the reset used for the Reset Remote Terminal Mode Code, except that command processing halts.

The Reset Command Register is illustrated in Figure 3-13.

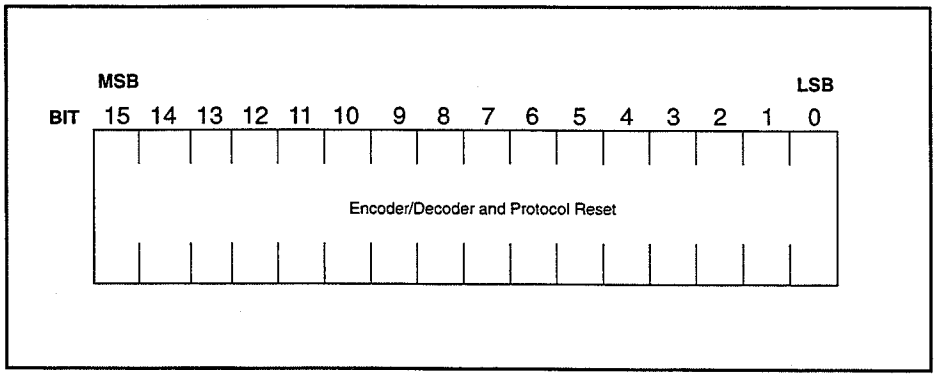


Figure 3-13 Reset Command Register

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.15 Register 13 — RT Timer Reset Register

Any write (such as, data = don't care) to Register 13's address location resets the RT Time Tag Timer to zero. The board's Remote Terminal Timer time-tags message transactions. The time-tag is generated from a free-running eight-bit timer of 64 microseconds resolution. The timer can be reset to zero simply by writing to Register 13. When the timer is reset, it immediately starts running. Register 13 is write-only.

The RT Timer Reset Register is shown in Figure 3-14.

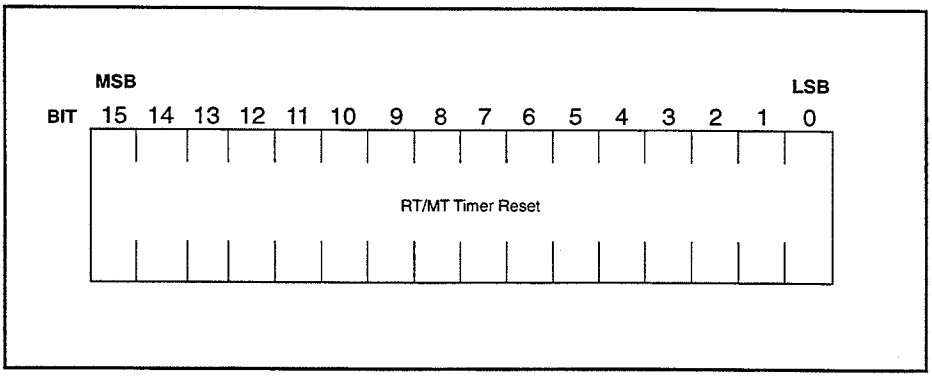


Figure 3-14 RT Timer Reset Register



NOTE: Registers 14 -17 are reserved for 1120 use only.

3.16 Register 14 — Monitor Terminal Control Register

The Monitor Terminal Control Register is shown in Figure 3-15 and the bits in this register are described in Table 3-9.

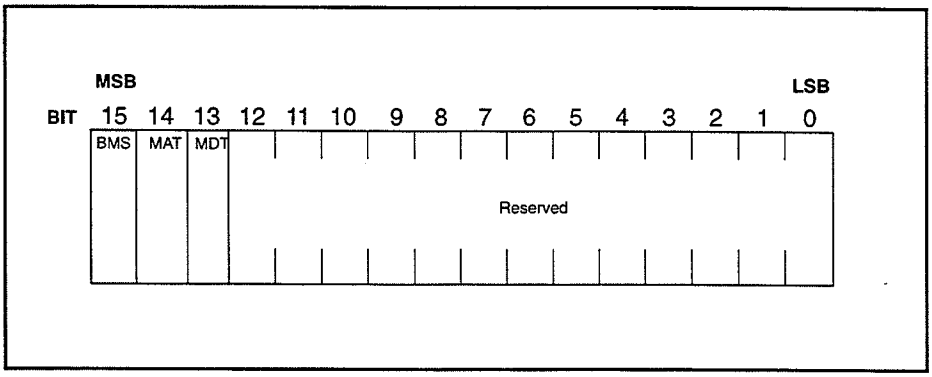


Figure 3-15 Monitor Terminal Control Register

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 3-9 Monitor Terminal Control Register

Bit	Function	Description
0-12	Reserved	
13	Monitor Declared Terminals (MDT)	When Bit 13 is set, the 1120 monitors only the remote terminals that are selected in Registers 16 and 17. If Bit 13 is not set, Bit 14 must be set. Bit 13 should be cleared for RT Mode operation.
14	Monitor All Terminals (MAT)	When Bit 14 is set, the 1120 monitors all remote terminal bus activity. If Bit 14 is not set, Bit 13 must be set. Bit 14 should be cleared for RT Mode operation.
15	Monitor Terminal Select (MTS)	The host sets Bit 15 to enable the 1120 Monitor Mode of operation. Bit 10 of Register 0 must also be "0" to enable the Monitor Mode. Bit 15 should be cleared for RT Mode operation.

3.17 Register 15 — Reserved



NOTE: Register 15 should not be accessed by the host.

3.18 Register 16 — Monitor Terminal Address Select RT (0-15)

Monitor selected remote terminal addresses (TA0-15). By setting the appropriate bit in Register 16 (Figure 3-16), the host can determine the remote terminals (RT 0 through RT 15) that the 1120 monitors. For example, by setting Bit 5 of Register 16, the host instructs the 1120 to monitor only the bus activity for RT 5. Because the Register 16 bits are not mutually exclusive, the host can monitor any number of different remote terminals by selecting the proper combination of bits.

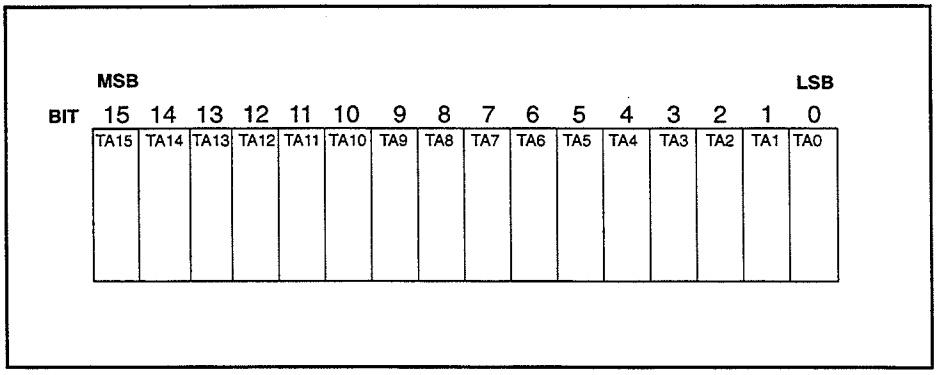


Figure 3-16 Monitor-Selected Remote Terminal Addresses 0-15

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.19 Register 17 — Monitor Terminal Address Select RT (16-31)

Monitor Selected Remote Terminal Addresses (TA16-31). By setting the appropriate bit in Register 17 (Figure 3-17), the host can determine the remote terminals (RT 16 through RT 31) that the 1120 monitors. For example, by setting Bit 5 of Register 17, the host instructs the 1120 to monitor the bus activity for only RT 21. Because the Register 17 bits are not mutually exclusive, the host can monitor any number of different remote terminals by selecting the proper combination of bits on Register 17 and Register 16.

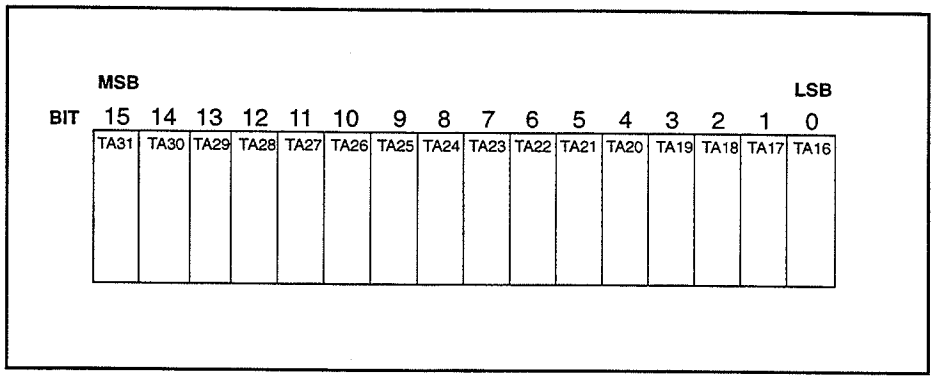


Figure 3-17 Monitor Selected Remote Terminal Addresses 31-16

3.20 Register 18 — Master Reset/Transceiver Built-In Test

The 112X hardware must be emulating RT30, SA0, Transmit and Mode Codes 3 and 19 for the Built-In Test (BIT) to be invoked. A READ to this register resets the Bus Controller and the Remote Terminal, clearing all internal registers.

When writing to Register 18, Bits 2 through 5 function as part of the BIT. The BIT verifies correct operation of the on-board MIL-STD-1553 bus transceivers. A hard-coded command word can be issued onto the bus from the hardware with transceivers disabled. The command can also be issued with transceivers enabled.

The test described above is repeated using a different command word and the results are verified to ensure correct operation of the on-board MIL-STD-1553 bus transceivers. The test also verifies functional operation of the hardware to receive valid bus commands. Register 4 provides the user with BIT results.

Any READ to Register 18 invokes a master reset to all sequencers, flip-flops and internal registers. A READ does not clear memory.

Figure 3-18 shows the Master Reset/Transceiver Built-In Test Register, and Table 3-10 describes the bits in this register.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

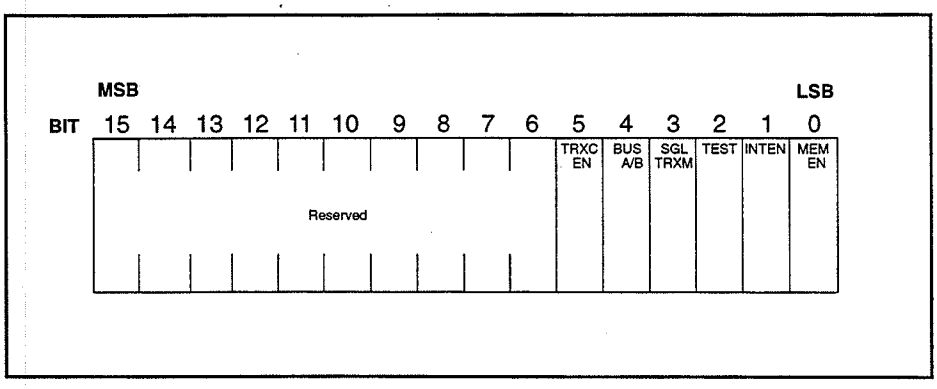


Figure 3-18 Master Reset/Transceiver Built-In Test Register

Table 3-10 Master Reset Transceiver Built-In Test Register

Bit	Function	Description
0	Memory Enable (MEMEN)	Setting bit 0 enables memory access from the PC.
1	Interrupt Enable (INTEN)	Bit 1 enables interrupts to the PC and prevents further access of the board. Set until relinquishing control of the interrupt level or returning control to the previous controlling device.
2	Test Pattern (TEST)	If Bit 2 is clear, Transmit Mode Command F413 (Transmit Bit)is initiated. When Bit 2 is set, Transmit Mode Command F403 is initiated.
3	Single Transmit (SNGTRXM)	When set, Bit 3 activates the Built-In Test. The Positive Triggered Single Word Transmit must be reset to 0 before re-transmit.
4	Bus A/B (BUSA/B)	Bit 4 selects either Bus A or B for the Built-In Test.
5	Transceiver Enable (TRXEN)	If cleared Bit 5 allows you to enable the transceiver. If set, Bit 5 allows you to perform control logic BIT, data generator, transmitting without going through transceivers.
6-15	Reserved	



NOTE: Bit 0 must be set for all WRITES to Register 18 to maintain board access.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.21 Register 19 — Interrupt Counter Read/Clear Register

The Interrupt Counter Read/Clear Register is shown in Figure 3-19, and the bits in this register are described in Table 3-11.

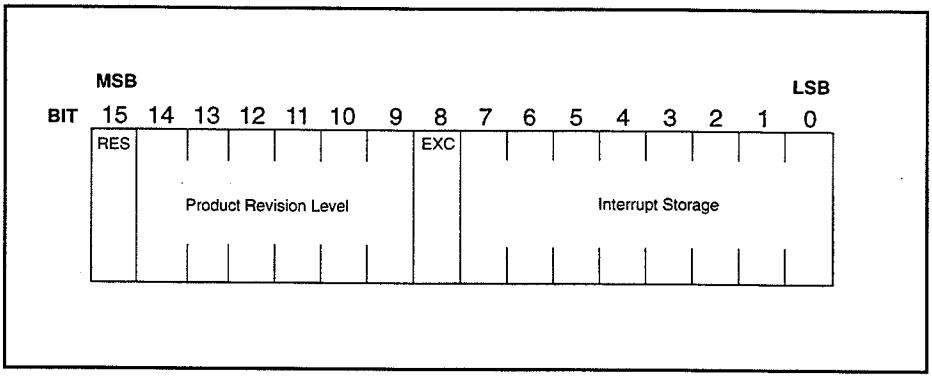


Figure 3-19 Interrupt Counter Read/Clear Register

Table 3-11 Interrupt Counter Read/Clear Register

Bit	Function	Description
0-7	Interrupt Storage Bits	Bits 0-7 store the number of interrupts held within the Standard Interrupt Queue. A read of any value to Register 19 will read the contents. A WRITE will clear the Exceed Bit and the counter (Bits 0-8).
8	Exceed Bit	When the 255 messages have been stored but not processed and another interrupt occurs, Bit 8 is set. The Exceed Bit can be cleared via a write to Register 19, a Master Reset, a system reset or a host reset.
9-14	Revision Level Bits	Bits 9-14 reflect the current revision level of the 112X product.
15	Reserved	

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

3.22 Register 20 — Autodecrement Interrupt Counter Register

Figure 3-20 shows the Autodecrement Interrupt Counter Register, and Table 3-12 describes the bits in this register.

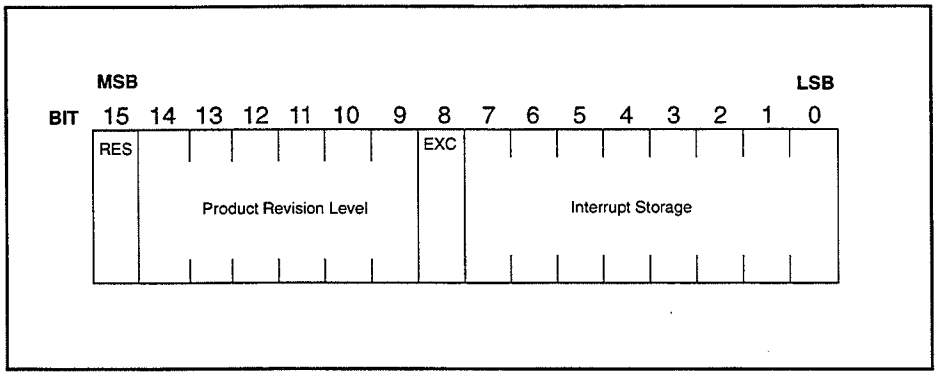


Figure 3-20 Autodecrement Interrupt Counter Register

Table 3-12 Autodecrement Interrupt Counter Register

Bit	Function	Description
0-7	Interrupt Storage Bits	Bits 0-7 store the number of interrupts held within the Standard Interrupt Queue. A read of any value to Register 20 will decrement the contents by one and return the current interrupt count. A WRITE will decrement the count field by one in both Registers 19 and 20.
8	Exceed Bit	When the 256th interrupt has been stored but not processed, the Exceed Bit is set. Bit 8 is cleared via a Master Reset, host system reset or WRITE to Register 19.
9-14	Revision Level Bits	Bits 9 -14 reflect the current revision level of the 112X product.
15	Reserved	

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

4.0 INTERRUPTS

4.1 Overview

The 112X has a powerful interrupt handling methodology that allows MIL-STD-1553B events to be defined as High Priority Interrupt, Standard Priority Interrupt or No Interrupt. The hardware switches select the interrupts levels for high and standard priority level.

The methods used to service Standard and High Priority Interrupts differentiate the two priority levels. High Priority Interrupts are recorded in the registers and are generally handled upon occurrence.

Standard Priority Interrupts are logged to a queue. Once interrupted, the user can read the interrupt from a queue. By setting Bit 0 of the High Priority Interrupt Register, the host is notified each time a Standard Priority Interrupt occurs. If Bit 0 is not set, the events are queued to the log without notifying the host.

The Standard Priority Interrupt Queue can handle up to 255 entries. The Interrupt Counter Register 19 efficiently counts the number of Standard Priority Interrupts queued.

The interrupt queue can be defined as circular or straight. With circular format queue, the earliest entry is overwritten, if not serviced when the queue fills up. With a straight format queue, the latest entry is overwritten. Interrupts are enabled, defined and handled by hardware switch banks, RT registers, and the application program being used.

4.2 Software-Programmable Registers

There are six 16-bit software programmable registers that control interrupts:

- Interrupt Queue Pointer Register (06),
- High Priority Interrupt Enable Register (07),
- High Priority Interrupt Status/Reset Register (08),
- Standard Interrupt Enable Register (09),
- Interrupt Counter Read/Clear Register (19) and
- Autodecrement Interrupt Counter Register (20)

The interrupt registers are programmed after all hardware switches are set and the board is installed. (See Chapter 3, Registers, for more detailed information on registers.)

4.3 Hardware-Selectable Switches

The hardware-selectable switches designate High and Standard Priority Interrupt Levels. The switches must be set before the 112X is installed in the host. (See Chapter 2 GETTING STARTED, for more information on switch selection.)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

4.4 Location Of Interrupt Service Routine

Standard and High Priority Interrupt handling routines, used for servicing and clearing queued or buffered interrupts, are defined by the user and located within the host memory. Figure 4-1 details the interrupt process.

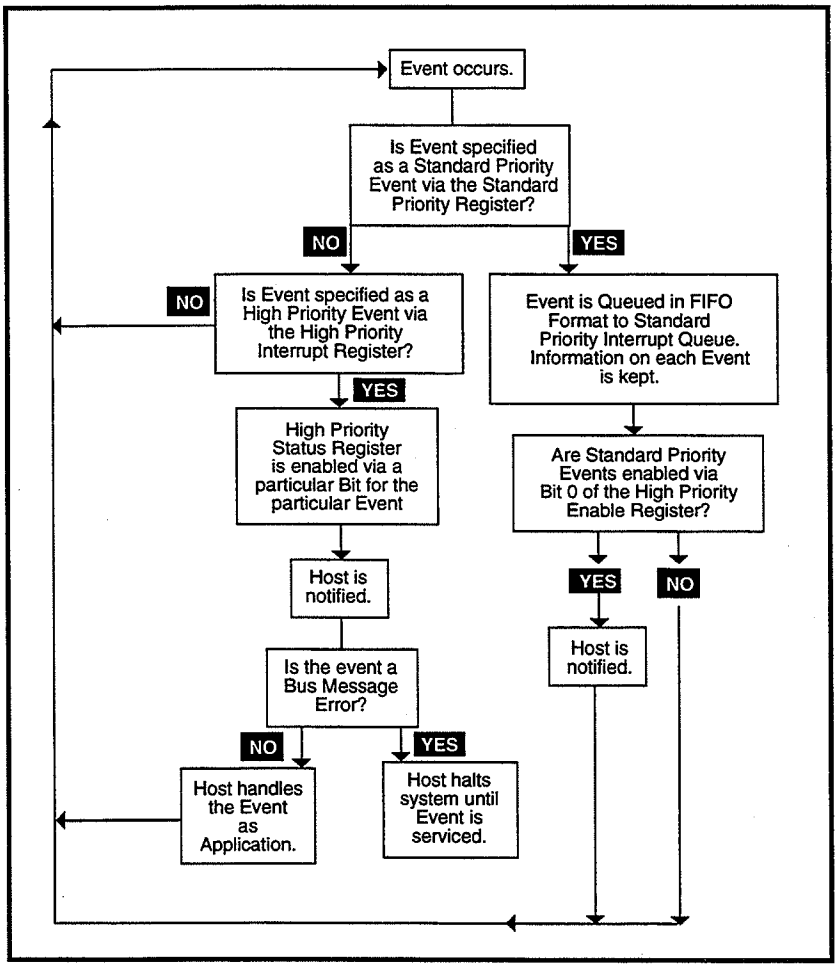


Figure 4-1 Interrupt Process

4.5 Interrupt Queue

The Standard Priority Interrupt Queue is initialized by the host and consists of a first-in/first-out queue of enabled interrupt events (Figure 4-2). Each entry in the Standard Priority Interrupt Queue consists of three 16-bit words:

- Interrupt Status Word
- Pointer:
  BC Mode - Bus Instruction Pointer
  RT Mode - Subaddress/Mode Code Response Block Pointer
  MT Mode - Monitor Command Pointer
- Tail Pointer

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

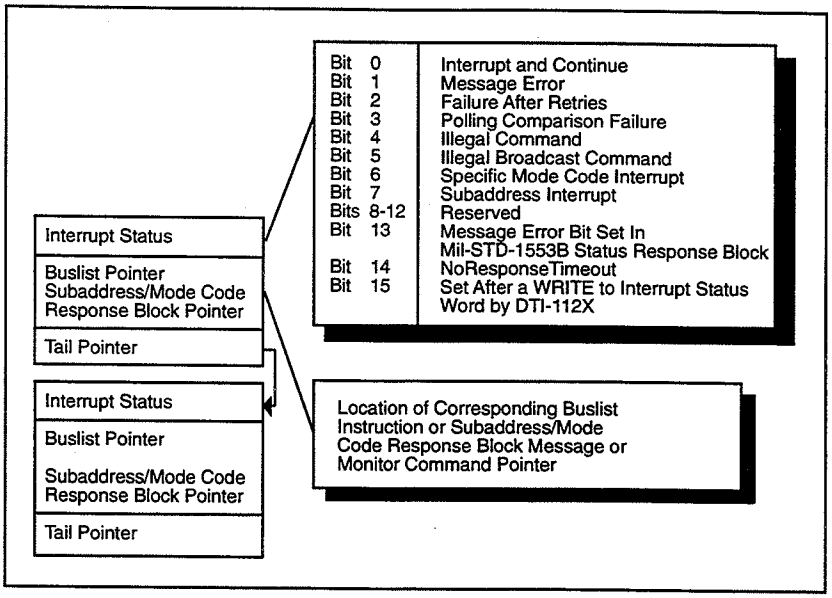


Figure 4-2 Standard Priority Interrupt Queue

4.5.1 Interrupt Status

The 16 bits of Status Information define the type of Standard Priority Interrupt that caused the interrupt.

4.5.2 Pointer

- BC Mode - The pointer indicates the location of the corresponding buslist instruction.
- RT Mode - The pointer indicates the location of the corresponding subaddress/mode code response block.
- MT Mode - The pointer indicates the location of the corresponding monitor command block.

4.5.3 Tail Pointer

The tail pointer contains the address of the next Standard Priority Interrupt entry in the queue. The subsequent entry does not have to be contiguous.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.0 BUS CONTROLLER MODE

5.1 Overview

In the Bus Controller Mode, the 112X contains a doubly-linked buslist structure and an internal register structure that provide a means of defining, storing and executing comprehensive buslist instructions. Under application control, the BC uses intermessage timing to emulate the instruction set and bus command timing without host intervention. The instructions specify an intermessage timer that delays the execution of subsequent instructions.

The 112X addresses a maximum of 32 remote terminals with up to 32 subaddresses each. Each buslist instruction, its size and method of execution is definable.

The bus controller also performs the following functions:

- Reads instructions from a buslist and retrieves the buslist sequentially
- Implements programmable Intermessage Timing, halting the execution of the next buslist instruction for up to 4080 microseconds
- Detects protocol and bus errors
- Initiates any Broadcast or Broadcast Mode Commands using RT Address 31
- Polls each RT status word and generates an interrupt, if specified
- Permits software-selectable interrupt generation
- Monitors data during RT-to-RT transfers
- Controls the Bus Retry Option, permitting the bus controller to re-send bus instructions on the same or opposite bus, up to four times

5.2 Command Block

In the Bus Controller Mode, the 112X can address up to 32 RTs. For each message transmitted over the 1553 bus, an associated Command Block exists. The Command Block contains a buslist, consisting of relevant message and RT status information, as well as programmable function bits that allow users to select functions and interrupts (see Figure 5-1).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

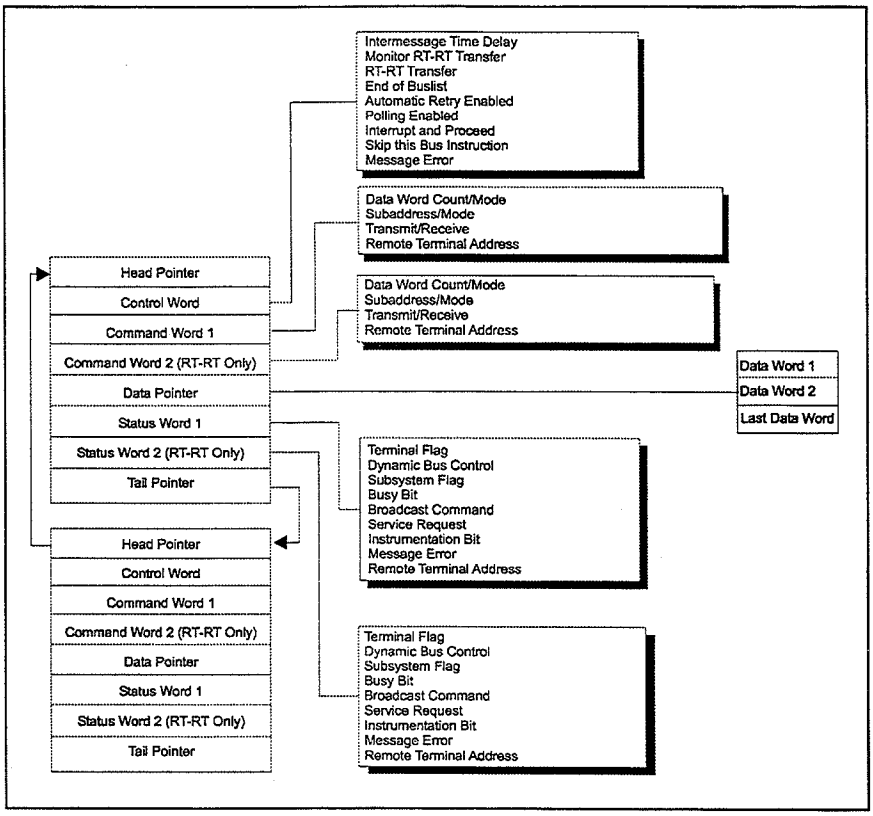


Figure 5-1 Bus Controller Structure

5.2.1 Head Pointer

The head pointer's 16 bits of information contain the address of the previous host-written buslist instruction. This linkage provides the capability to loop through a list of bus instructions.

5.2.2 Control Word

The Control Word contains bit-programmable functions, allowing individual functions to be selected for each message (see Figure 5-2). The Control Word's 16 bits regulate the following features:

- Intermessage Timing
- RT-to-RT Transfers
- End of Buslist
- Retry Capability
- Polling
- Interrupt and Continue
- Skipping Instructions
- Message Errors

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

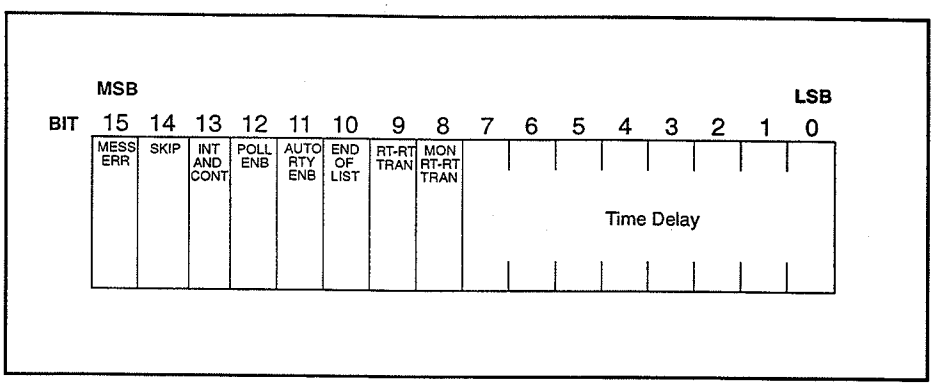


Figure 5-2 Control Word Register

Table 5-1 summarizes the functions of the bits in the Control Word Register and the following paragraphs give more detail for bits 0-7, 11-12, and 15.

Table 5-1 Control Word Register

Bit	Function	Description
0-7	Time Delay	The CPU sets Bits 0-7 and causes the 112X to delay the specified time between sequential message starts.
8	Monitor RT-RT Transfer (MON RT-RT TRAN)	Set by the CPU, Bit 8 indicates the 112X should receive and store the data in this RT-RT transfer, beginning at the location indicated by the data pointer.
9	RT-to-RT Transfer (RT-RT TRAN)	Set by the CPU, Bit 9 indicates the bus instruction is an RT-to-RT Transfer.
10	End of Buslist (END OF LIST)	If enabled, Bit 10 indicates that upon completion of the current message the 112X will halt and assert a High-Priority Interrupt. The interrupt must also be enabled in the High-Priority Interrupt Enable Register.
11	Auto Retry Enable (AUTO RTY ENB)	When set, the Auto Retry Bit, governed by global parameters in the Control Register, is enabled for this message.
12	Polling (POLL ENB)	If Bit 12 is set, polling is enabled.
13	Interrupt and Continue (INT AND CONT)	If Bit 13 is set and this block is addressed, a Standard Interrupt is asserted without halting operation.
14	Skip	If Bit 14 is enabled with a value of one, the 112X skips this Command Block and executes the next. If the Intermessage Timer has been enabled, the timer value will expire before proceeding to the next bus message.
15	Bus Message Error (BUS MSG ERR)	If a message error is detected during the execution of this buslist instruction, Bit 15 is set.

NOTE: Bit 13 must be enabled by setting Bit 0 of Register 9.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

BITS 0-7 — TIME DELAY

Regardless of the value in the Time Delay field (including zero), the 112X will at least delay the minimum 4 µs intermessage gap time as specified in MIL-STD-1553B. The timer is enabled by having a non-zero value in this bit field.

Timer resolution is 16 microseconds. For example, if a given message requires 116 µs to complete (including the minimum 4µs Intermessage Gap), the value in the Time Delay field must be at least 000010000 (8 x 16 µs = 128 µs) to provide an intermessage gap greater than the 4 µs minimum requirement (see Figure 5-3).



NOTE: If the timer is enabled and the Skip Bit is set, the timer provides the programmed delay before proceeding. If the message duration exceeds the Time Delay, the message is completed as if the timer is not enabled.

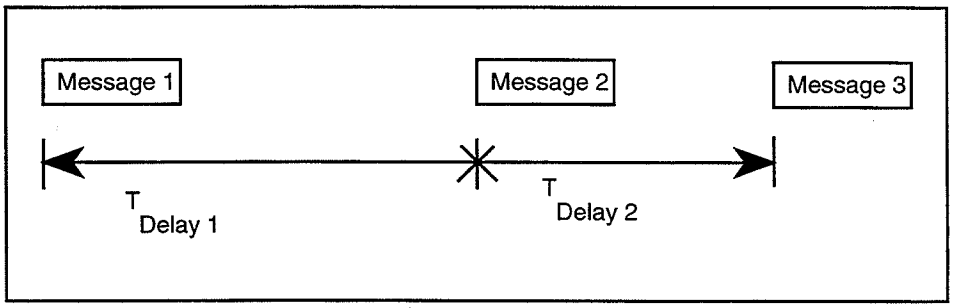


Figure 5-3 BC Timing Delays

BIT 11 — AUTO RETRY ENABLE (AUTO RTY ENB)

The Auto Retry can be activated when any of the following events occur:

- Busy Bit set in Remote Terminal Status Response
- Message Error, indicated by the RT Status Response
- Response Time Out
- Message Error detected by the Bus Controller

One to four retries are programmable on the same or opposite bus.

BIT 12 — POLLING (POLL ENB)

The contents of the least significant eleven bits of the RT's Status Word are anded to the value of the Polling Compare Register.

During a typical polling scenario, the Bus Controller instructs remote terminals to transmit their Status Words, which alerts the host if a bit is set in any RT Status Word during normal message transactions. The 112X enables the host to initialize a chain of Command Blocks with the Command Word's Polling Enable Bit.

A programmable Polling Compare Register (PCR) is provided. In the Polling Mode, the Remote Terminal response is compared to the PCR' contents.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Program the PCR by setting the PCR Bits corresponding to the RT's 1553 Status Word Bits being compared. If the bits match and Polling is enabled in the BC Command Block Control Word and in the Standard Interrupt Enable Register, a Polling Comparison Interrupt is generated. Examples of both a bit match and no bit match are shown below:

EXAMPLE 1 — NO BIT MATCH IS PRESENT.

PCR	00000000001
RT'S 1553 Status Word Response	00000100010
Result:	No Polling Comparison Interrupt

EXAMPLE 2 — BIT MATCH IS PRESENT.

PCR	00100100000
RT's Status Word Response	00000100000
Result:	Polling Comparison Interrupt

BIT 15 — BUS MESSAGE ERROR

The Bus Controller checks for errors on each message transaction. In addition, the BC compares the RT Command Word Addresses to the incoming Status Word Addresses. The BC monitors for response time-out and checks data and Control Words for proper format according to MIL-STD1553B. Illogical Commands include incorrectly formatted RT-RT Command Blocks.

Bus Message Errors can be enabled as High Priority Interrupts, Standard Priority Interrupts or No Interrupt. If Bus Message Error is enabled as a High Priority Interrupt, the 112X halts operation when a Bus Message Error occurs until the interrupt is serviced. If a Bus Message Error is enabled as Standard Priority, the data is queued.

5.2.3 Command Word One

Command Word One is used for all MIL-STD-1553B transfers. It is initialized by the PC-AT host and contains the first Command Word corresponding to the Command Block's message transfer. The first command of a Remote-to-Remote message type is a receive command.

5.2.4 Command Word Two

Command Word Two is initialized by the host and contains the second Command Word for RT-to-RT Transfers. Command Word Two contains the transmitting RT address, subaddress, and the word count. Command Word Two is not used in single RT Transfers.

5.2.5 Data Pointer

The host-initialized Data Pointer contains the starting location in RAM for the Command Block's message (see Figure 5-4).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

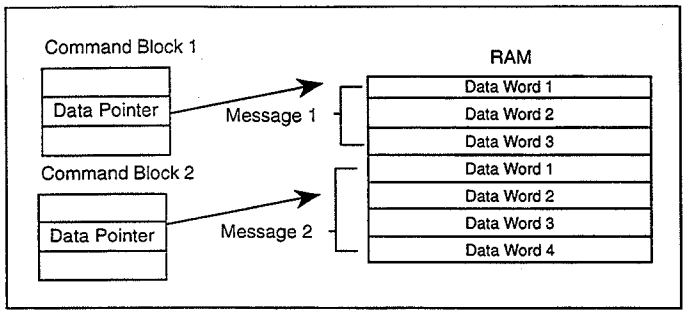


Figure 5-4 Contiguous Data Storage

5.2.6 Status Word One

Status Word One is stored by the 112X and contains the entire Remote Terminal Status Response. Status Word One is loaded in one of the following cases:

- An RT Response Status for a BC-to-RT Transfer (Master-to-Remote transfer) or a RT-to-BC Transfer (Remote-to-Master Transfer) occurs
- A Status Word of MIL-STD-1553B Mode Code Transfer occurs
- An RT Transmit Status for an RT-to-RT Transfer occurs

5.2.7 Status Word Two

The Status Word is stored by the 112X and contains the receiving Status in RT-to-RT Transfers. The location is not used for transfers involving only one Remote Terminal.

5.2.8 Tail Pointer

The 16 bits of the Tail Pointer contain the location of the next sequential buslist instruction.

5.3 Linking Command Blocks

The 112X Command Blocks are doubly-linked. In a doubly-linked Command Block structure, pointers delimit each Command Block to the previous and successive blocks (see Figure 5-5). This linking feature eases multiple message processing tasks and supports message scheduling by looping through a series of transfers at a pre-determined cycle time.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

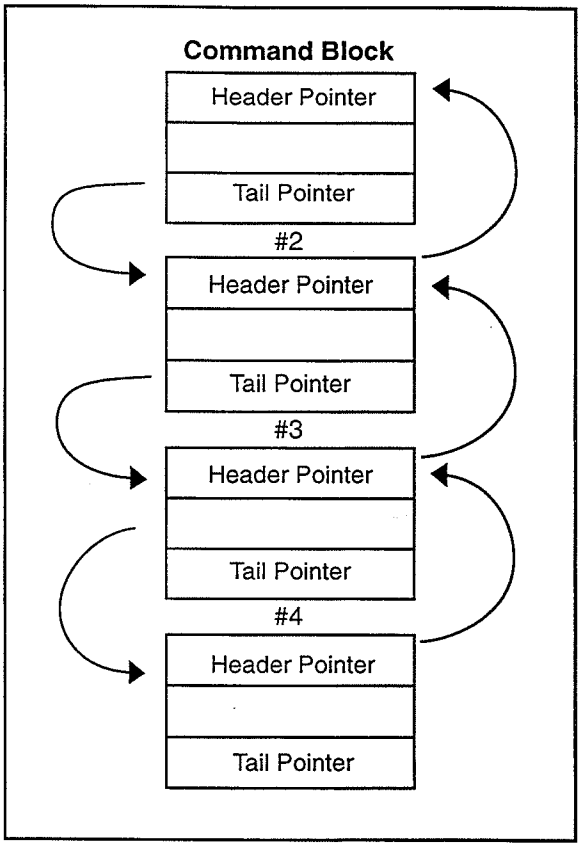


Figure 5-5 Command Block Linking

5.4 BC Functional Operation

The host determines the first Command Block by setting the initial starting address in the Current Command Block Register. Once set, the 112X updates the current Command Block Register with the next Command Block Address. The 112X then executes the sequential Command Blocks and counts out programmed message delays until it encounters the last Command Block listed. The last Command Block is indicated by the End of List Bit in the Control Word. Interrupts are asserted when enabled events occur.

The following paragraphs provide a general description of 112X message processing in the BC Mode.

Before message processing may occur, the user starts BC operation by writing a '1' to Register 0, Bit 0. After BC startup, the 112X performs a memory read from the third location of the Command Block.

5.4.1 BC-to-RT Transmit

In BC-RT Transmit, the 112X first transmits the Command Word. The 112X then performs a Data Word Memory Read, starting at the Data List Pointer Address, incremented for each successive word. The 112X transmits the Data Word to an RT. Data Word Memory Reads and Transmissions continue until all Data Words are transmitted. Next, the 112X receives a Status Word from the RT and writes the Status Word to the sixth location of Command Block.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

5.4.2 RT-to-BC Transmit

In RT-BC Transmit, the 112X first transmits the Command Word. The 112X then receives a RT Status Word and writes it to the sixth location of the Command Block. The 112X receives the first Data Word from the RT and writes the Data Word to memory, beginning at the Data List Pointer Address, incremented for each successive word. Data Word Receptions and Memory Reads continue until all Data Words are received.

5.4.3 RT(B)-to-RT(A) Transmit

In RT(B)-to-RT(A) Transmit, the 112X transmits Command Word 1 to RT(B) and performs a Command Word 2 Read from the fourth location of the Command Block. The 112X then transmits Command Word 2 to RT(A). The 112X receives the RT(A) Status Word and writes the word to the sixth location in the Command Block.

The 112X receives the first Data Word. If the 112X is enabled to monitor the RT-to RT message, the 112X performs a Data Word Memory Read and writes the Data Word to memory, beginning at the Data List Pointer Address, incremented for each successive word. Data Word Receptions and Memory Reads continue until all Data Words are received.

Next, the 112X receives the RT Status Word from RT(B) and writes the Status Word to the seventh location in the Command Block.

5.5 Interrupts in the Bus Controller Mode

Table 5-2 lists the causes of High Priority and Standard Priority Interrupts for the 112X when in the BC Mode:

Table 5-2 Causes of Interrupts

High Priority Interrupts	Standard Priority Interrupts
Halt Execution	Command Execute
Illogical Command	Retry Failure
Subsystem Fail	Status Exception
Message Error	Message Errors

5.6 Mode Commands in BC Mode

As a Bus Controller, the 112X emulates all MIL-STD-1553 Mode Codes. For the 112X to emulate Mode Codes, Bit 13 of the Control Register (0) must be reset. Otherwise, Subaddress 31 is interpreted as a data subaddress. See Section 6.0 REMOTE TERMINAL MODE for more information on Mode Codes.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

6.0 REMOTE TERMINAL MODE

6.1 Overview

In the Remote Terminal Mode, the 112X can emulate a single MIL-STD-1553B remote terminal with up to 30 data subaddresses and two mode subaddresses or 31 data subaddresses and one mode subaddress (Subaddress 0). Changing remote terminal emulation from one RT number to another RT number is software-selectable.

By using the 32 K-word Dual-Port RAM, the 112X can receive, store, timetag and count approximately 1000 bus messages of 32 data words each.

The RT Mode can buffer up to 128 receive messages and 128 transmit messages to the same subaddress without host intervention. The 112X stores data per subaddress in a first-in/first-out methodology with a 64-microsecond timetag resolution. As each valid bus message is received, a remote terminal index counts the number of storage buffers. If the buffers for a particular subaddress are full, only the last message buffer is overwritten, preventing data corruption of messages previously stored in the buffer.

The Remote Terminal also performs the following functions:

- Generates an interrupt when a subaddress or mode code is accessed or a user-defined number of messages are reached
- Detects message errors
- Enables specific subaddresses/mode codes

If multiple RT emulation is required, additional boards can be installed in a single PC-AT host.

6.2 Host-Programmable Registers

Two 16-bit software-programmable registers control the remote terminal: the Remote Terminal Address Register (10), which sets the RT address, and the Remote Terminal Timer Register (13), which resets the Time Tag Timer.

The RT registers are programmed after all hardware switches are set and the board is installed. (See Section 3.0 REGISTERS for details.)

6.3 RT Subaddress Block

When an RT receives a valid instruction from the bus controller, a Memory Read is initiated to read the RT's subaddress response block. The response block contains control words and pointers that determine how data transfers are performed (Figure 6-1).

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

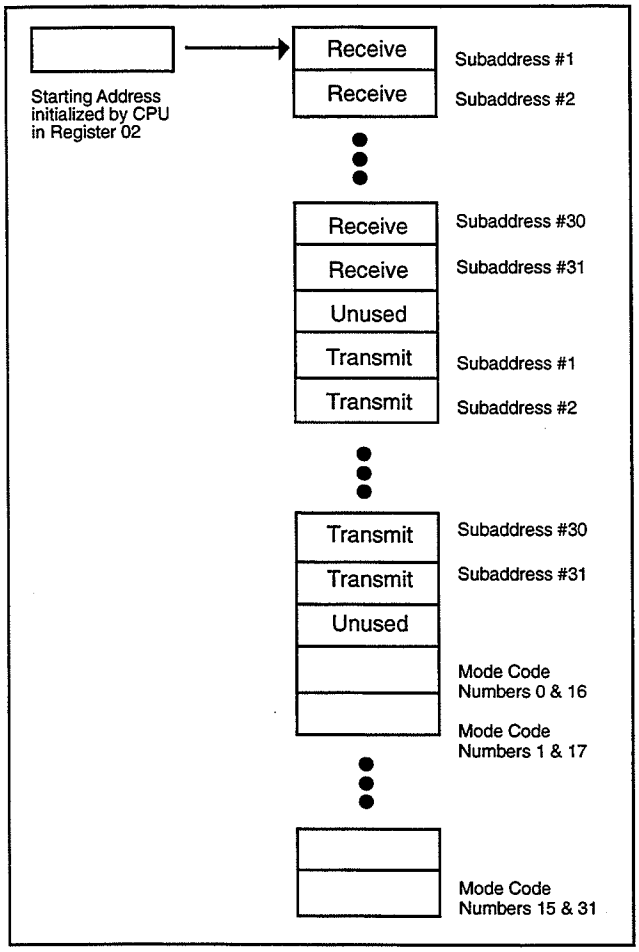


Figure 6-1 RT-Subaddress Blocks

The RT subaddress block consists of three words—a control word, a message status list pointer, and a data list pointer—and a reserved fourth word (see Figure 6-2).

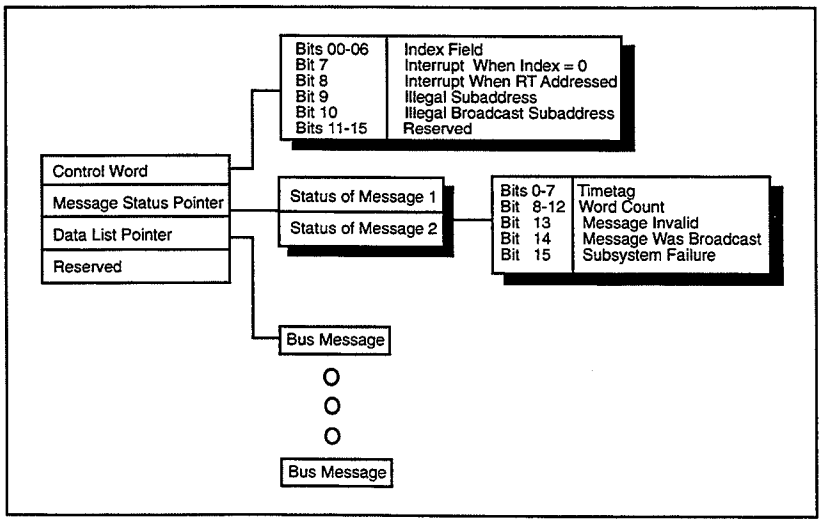


Figure 6-2 Remote Terminal Structure

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

6.3.1 Control Word

The 16-bit control word selects or disables message transfers and counts the number of messages queued to each subaddress (Figure 6-3).

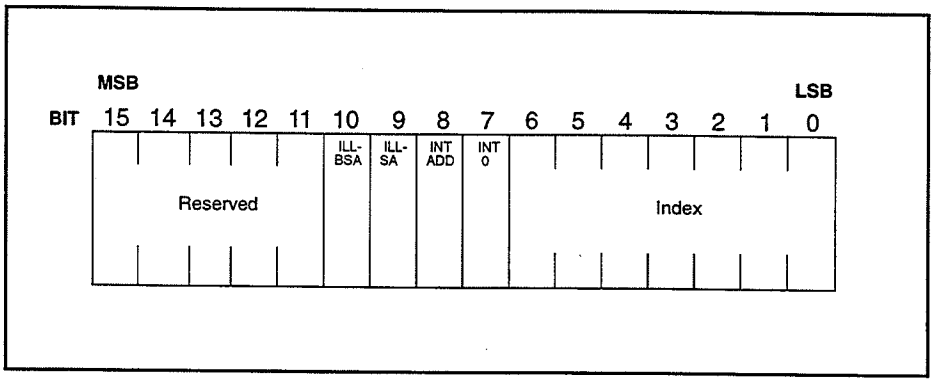


Figure 6-3 RT Subaddress Control Word

Table 6-1 describes the bits in the RT Subaddress Control Word, and the following paragraphs give more detail for bits 0-6 and 9.

Table 6-1 RT Subaddress Control Word

Bit	Function	Description
0-6	Index	Bits 0-6 count the number of messages stored per subaddress.
7	Interrupt When Index is Equal to Zero (INT0)	Bit 7 works in conjunction with Bits 0-6 to define a subaddress event interrupt. Bits 0-6 and 7 notify the host that a particular RT Subaddress is full because the index is decremented from '1' to '0' when this message is transacted. If Bit 6 changes from '1' to '0' and Bit 7 equals '1', a Subaddress Event Interrupt occurs.
8	Interrupt Upon Valid Command Received (INTREC)	If Bit 8 is enabled, a Subaddress Event interrupt occurs. Each time the subaddress is addressed and Bit 8 equals '1', a Subaddress Event Interrupt Standard Priority Interrupt occurs.
9	Illegal Subaddress (ILLSA)	If Bit 9 is enabled, the subaddress will not respond to a bus message sent to it.
10	Illegalize Broadcast Subaddress (ILLBSA)	If Bit 10 is enabled, the subaddress is not enabled for broadcast. If the Standard Interrupt Enable Register (09), Illegal Broadcast Command Received events are enabled, an Illegal Broadcast Command Received Interrupt occurs each time a broadcast message references this subaddress. Bit 10 indicates all RT subaddresses that were issued a broadcast but are not enabled to receive broadcasts.
11-15	Reserved	

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

BITS 0-6 — INDEX

A maximum of 128 messages can be stored per subaddress before overwriting occurs. Each time a message is stored to a subaddress, the index of the subaddress is decremented by one until index = 0. The index is not incremented or decremented if an invalid command word is received.

A maximum of 128 messages can be stored for RT transmits and up to 128 messages can be stored for RT receives per subaddress; however, up to 1000 32-word messages can be stored by all subaddresses combined. As each valid message is completed, Bits 0-6 are decremented by one.

BIT 9 — ILLEGAL SUBADDRESS.

If the Illegal Command Received Interrupt is enabled in the Standard Interrupt Enable Register (09) and Bit 9 is set, an Illegal Command Received Standard Priority Interrupt occurs each time a message involving this subaddress is transacted.

Bit 9 can be used to determine if the bus message should have been sent to another RT subaddress or if the RT subaddress should be valid. If the subaddress is supposed to be valid, the RT configuration is incorrect.

6.3.2 Message Status Pointer

The 112X uses the message status pointer, the second word within the response block, as a starting address for the message status list. The pointer is incremented by one with each message status word write. If the control word index already equals '0', the message status list pointer is not incremented and the previous message status word is overwritten.



NOTE: A message status word is also written and the proper pointer incremented when the 112X detects a message error.

6.3.3 Data List Pointer

The data list pointer is the third word within the subaddress response block. The 112X stores data in RAM beginning at the address indicated by the data list pointer. The data list pointer is updated at the end of each successful message with the next message's starting address, excluding the following circumstances:

- If the message is erroneous, the data list pointer is not updated. The next message over writes any data corresponding to the erroneous message.
- If the index already equals 0 upon receiving a message, the data list pointer is not incremented and data from the previous message is overwritten.

If the bus controller sends the RT a Receive Command, the RT buffers data into the indexed queue. The queue stores up to 128 buffers per subaddress. Each buffer is capable of storing up to 32 data words. When a message is placed in the RT's queue, the message is automatically time-tagged and the index is decremented.

To count the number of words received, subtract the current index from its original value.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

EXAMPLE

If the current pointer equals 100 and the original pointer was 98, two words have been received (100 - 98 = 2 words received).



NOTE: Only data buffers with valid status condition are queued; however, if the RT recognizes the command word (whether data is valid or invalid), a message status word is generated and the index is decremented.

The number of messages held in each queue is easily monitored. To provide maximum data integrity when a buffer fills up, only the last message of the full queue is written. The queue can be re-written when all of the messages in the queue are serviced, if the original message status pointer and data list pointer are re-written to the second and third words of the response block. An interrupt can be generated when the buffer becomes full.

6.4 Message Status Word

Each message the 112X transacts has a corresponding message status word that is pointed to by the message status list pointer. The message status word allows the CPU to evaluate the message's validity, determine the word count and calculate the approximate time frame necessary to transact the message (Figure 6-4).

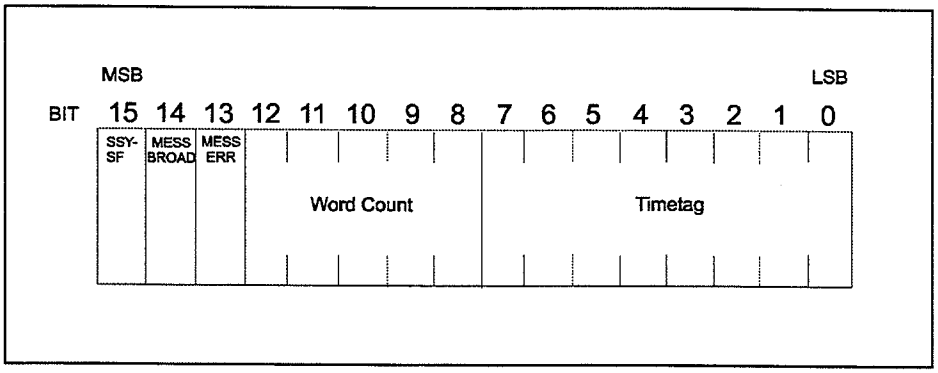


Figure 6-4 Message Status Word Bits

The bits in the Message Status Word are described in Table 6-2.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 6-2 Message Status Word Bits

Bit	Function	Description
0-7	Time Tag	The 112X writes the internally generated timetag to Bits 0-7 after message completion with a resolution of 64 microseconds. (See Register 13.) If the timer reads "2", it indicates the message was completed 128 to 191 microseconds after the timer started.
8-12	Word Count	Bits 8-12 indicate the number of words in the message and reflect the word count field in the command word. If the message contains a different number of words than the word count field, the Message Error Flag is triggered. If the message contains too many words, the words are withheld from RAM. If the actual word count is lower than it should be, the Message Error Bit in the 1553 status word is set.
13	Message Error (MESSERR)	Bit 13 indicates a message is invalid because of improper synchronization, bit count, word count and Manchester Error.
14	Broadcast Message (BROADMESS)	Bit 14 indicates the corresponding broadcast message was received.
15	Subsystem Failure (SSYSF)	Bit 15 indicates SSYSF was asserted just before the status word transfer to memory occurred. Bit 15 is also set when the user sets Bit 13 of Register 10.

6.5 RT Mode Code

Frequently, the instruction sent to an RT is in the form of a mode code. Mode codes are handled similarly to subaddress transactions. Both use four-word descriptors, which reside in the response block, to allow the host to program their operational mode.

Of 32 possible mode code address combinations, the 112X supports those listed in Table 6-3. The following paragraphs contain more details about these supported combinations.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 6-3 Mode Codes

Associated Mode Code	Function	Data Word	Broadcast
0	Dynamic Bus Control	No	No
1	Synchronize	No	Yes
2	Transmit Status Word	No	No
3	Initiate Self Test	No	Yes
4	Transmitter Shutdown	No	Yes
5	Override Transmitter Shutdown	No	Yes
6	Inhibit Terminal Flag Bit	No	Yes
7	Override Inhibit Terminal Flag Bit	No	Yes
8	Reset Remote Terminal	No	Yes
16	Transmit Vector Word	Yes	No
17	Synchronize with Data Word	Yes	Yes
18	Transmit Last Command	Yes	No
19	Transmit BIT Word	Yes	No
20	Selected Transmitter Shutdown	Yes	Yes
21	Override Selected Transmitter Shutdown	Yes	Yes

- **Dynamic Bus Control — #00000**. The Dynamic Bus Control Mode Code is accepted automatically if the Dynamic Bus Control Enable Bit in the Remote Terminal Address Register is set. Setting the Dynamic Bus Control Acceptance Bit in the 1553 status word and 112X Status Register confirms the mode code acceptance. A High Priority Interrupt is also asserted, if enabled. If the Dynamic Bus Control Bit is not set, the 112X rejects the Dynamic Bus Control.
- **Synchronize (without data word) — #00001**. If the Interrupt When Mode Code Received Bit is enabled in the Mode Code #00001 Control Word, the 112X asserts an interrupt when the mode code is received.
- **Transmit Status Word — #00010**. The 112X automatically transmits the 1553 status word corresponding to the last message transacted.
- **Initiate Self-Test** Initiate Self-Test— **#00011**. The 112X automatically starts its BIT routine. When the test is completed, any enabled interrupts are asserted and the BIT Word Register is updated.
- **Transmitter Shutdown — #00100**. The 112X disables the channel opposite the channel on which the command was received.
- **Override Transmitter Shutdown — #00101**. The 112X enables the channel previously disabled in Transmitter Shutdown.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

- **Inhibit Terminal Flag Bit — #00110**. The 112X inhibits the Terminal Flag from being set in the Status Word.
- **Override Inhibit Terminal Flag Bit — #00111**. The 112X disables the Terminal Flag Inhibit.
- **Reset Remote Terminal — #01000**. The 112X automatically resets the encoder, decoders and protocol logic.
- **Transmit Vector Word — #10000**. The 112X transmits the Vector Word from the location addressed by the data list pointer in the mode code response block.
- **Synchronize (with Data Word) — #10001**. Upon receiving this mode code, the 112X stores the associated data word.
- **Transmit Last Command — #10010**. The 112X transmits the last command executed and the corresponding status word.
- **Transmit Bit Word — #10011**. The 112X transmits BIT information from the BIT Register.
- **Selected Transmitter Shutdown — #10100**. Upon receiving this mode code, the 112X stores the associated data word.
- **Override Selected Transmitter Shutdown — #10101**. Upon receiving this mode code, the 112X stores the associated data word.



NOTE: Mode Codes 9-15 and 22-31 are reserved for future expansion of MIL-STD-1553B.

6.6 Mode Code Response Block

Sixteen mode code response blocks are used for mode code operations with each block handling two mode codes. The mode code response blocks, numbered sequentially from 0 to 15, correspond to Mode Codes 0 to 15 without data words and Mode Codes 16 to 31 with data words. For example, Mode Codes 0 and 16 correspond to Response Block 0 and Mode Codes 1 and 17 correspond to Response Block 1. The mode code response block is appended to the subaddress response block, beginning at 0100 *hex* (256D) of the 320-word RT response block.

The 112X autonomously supports all mode codes without data words by executing the specific function and transmitting the 1553 status word. The subsystem provides the status word for the mode codes with data words.

An interrupt can be asserted on all mode codes upon completion of the mode command. To set an interrupt, set the appropriate bit in the control word.

Each mode code block contains four words — a control word, message status pointer, data list pointer and a reserved word.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

6.6.1 Control Word

The mode code control word consists of 16 bits of response information that select or disable message transfers and count the number of messages queued to each mode code with data (Figure 6-5).

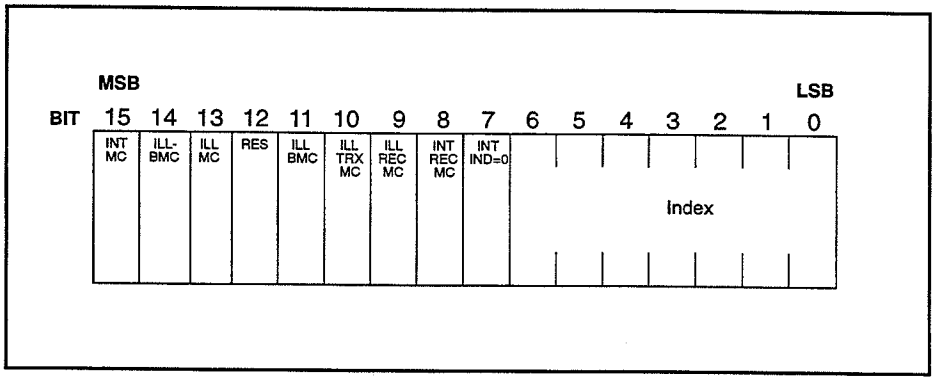


Figure 6-5 Mode Code Control Word

The bits in the Mode Code Control Word are described in Table 6-4.

Table 6-4 Mode Code Control Word

Bit	Function	Description
0-6	Index	Bits 0-6 apply only to mode codes with data. The bits count the number of messages stored per mode code with data. A maximum of 128 messages can be stored per mode code before overwriting occurs. The initial value per mode code is user-definable at up to 127. Each time a message is stored to a mode code, the corresponding index is decremented by one. The index is not decremented if an invalid command is received.
7	Interrupt When Index Is Equal To 0. (INT=0)	Bit 7 is used in conjunction with Bits 0-6 to define a Subaddress Event Interrupt event. When Bits 0-6 change from '1' to '0' and Bit 7 equals '1', a Subaddress Event Interrupt occurs. The host is notified when all the message buffers of a particular mode code with data block are full, which maximizes the integrity of stored data.
8	Interrupt When Mode Code (With Data Word) Is Transacted	Bit 8 allows an interrupt to be generated when a mode code with data is transacted. When the RT transacts the mode code with data, a Subaddress Event Interrupt occurs. By enabling Bit 8, monitoring activity to and from a specific subaddress is simplified.
9	Illegalize Receive Mode Code (with Data Word)	If Bit 9 is enabled, the RT ignores a receive mode code with data word sent to its mode subaddress. If Illegal Command events are enabled via the Standard Interrupt Enable Register (No. 09) and Bit 9 is set, an Illegal Command Standard Priority Interrupt occurs when the RT transacts the mode code with data. Bit 9 allows for mode code validation.
10	Illegalize Transmit	If Bit 10 is enabled, the RT ignores a Transmit Mode

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 6-4 Mode Code Control Word		
Bit	Function	Description
	Mode Code (with Data Word)	Code with Data Word sent to its mode subaddress. If Illegal Command events are enabled via the Standard Interrupt Enable Register (No. 09) and Bit 10 is set, an Illegal Command Standard Priority Interrupt occurs when the RT transacts the mode code .
11	Illegalize Broadcast Mode Code (with Data Word)	If Bit 11 is enabled, the RT ignores a Broadcast Mode Code with data word sent to it. If Illegal Broadcast Command Received events are enabled via the Standard Interrupt Enable Register and Bit 11 is set, an Illegal Broadcast Command Received Standard Priority Interrupt occurs when the RT transacts the mode code with data and Bit 11 equals '1'. Bit 11 notifies the user of attempts to transact the Broadcast Mode Code.
12	Reserved	
13	Illegalize Mode Code (without Data Word)	If Bit 13 is enabled, the RT ignores the mode code without data. If Illegal Command events are enabled via the Standard Interrupt Enable Register and Bit 13 is set, a Subaddress Standard Priority Interrupt occurs when the RT transacts the mode code without data.
14	Illegalize Broadcast Mode Code (without Data Word)	If Bit 14 is enabled, the RT ignores the Broadcast Mode Code. If Illegal Broadcast Command Received events are enabled via the Standard Interrupt Enable Register and Bit 14 is set, an Illegal Broadcast Command Received Standard Priority Interrupt occurs when the RT transacts the mode code .
15	Interrupt on Reception of Mode Code (without Data Word)	If Bit 15 is enabled, the RT will interrupt when the mode code is transacted. If Bit 15 is set, a specific mode code interrupt occurs when the RT transacts the mode code without data.

6.6.2 Message Status Pointer

The 16-bit message status pointer links RT status information to each mode code with data word message and specifies the location of the RT status information.

6.6.3 Data List Pointer

When a bus message is sent to an RT, the message is placed in the first-in first-out message queue for that particular mode code with data. Any associated data words are linked via the data list pointer to the corresponding bus message.

The data list pointer contains a 16-bit address of the next available location for storing incoming data. If the index field for a particular mode code with data equals 0, the data list pointer is not incremented. The last stored data is written and re-written.

If the index field for a mode code with data does not equal 0, the data list pointer is updated with the next available data location at the end of the message. If the data is erroneous, the pointer is not changed.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

7.0 MONITOR TERMINAL MODE

7.1 Overview



NOTE: This section applies only to the 1120 Board.

The monitor terminal structure is based on a command block structure and internal, host-programmable registers. The 1120 board may be programmed to act as a monitor terminal. The host uses Registers 14, 16 and 17 in conjunction with Register 0 to program the 1120 to monitor one RT, any combination of RTs or all RTs. The monitor terminal also timetags messages and stores approximately 800 messages of 32 data words each, without host intervention.

In the Monitor Terminal Mode, the 1120 checks all monitored messages for errors. Detectable errors include the following:

- Response Time-out
- Long and Short Word Count
- Bit Count and Manchester II
- Parity
- Data Contiguity

7.2 Command Block

Each monitored RT message transacted over the 1553 bus has an associated command block, set up by the host. The command block contains all relevant message and RT status information. The command block also houses programmable function bits that allow the user to select functions and interrupts.

The terminal monitor command blocks are linked to each other in a loop fashion. In the linked command block structure, a pointer delimits each command block to the successive block, allowing for more efficient space allocation (Figure 7-1).

Each block contains one data buffer. The user reserves space in memory, before monitoring begins, for the maximum number of data words appearing in any message. If sufficient memory is not allocated for the data buffers, some data words may be overwritten by sequential messages.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

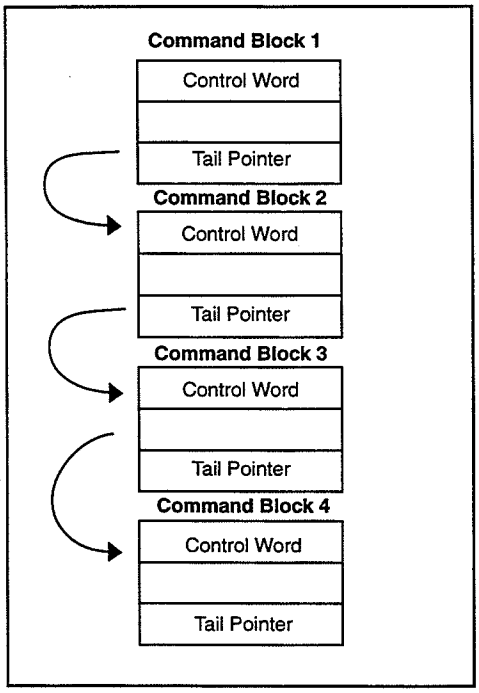


Figure 7-1 Monitor Terminal Command Block Pointers

The monitor terminal command block consists of seven 16-bit words (Figure 7-2).

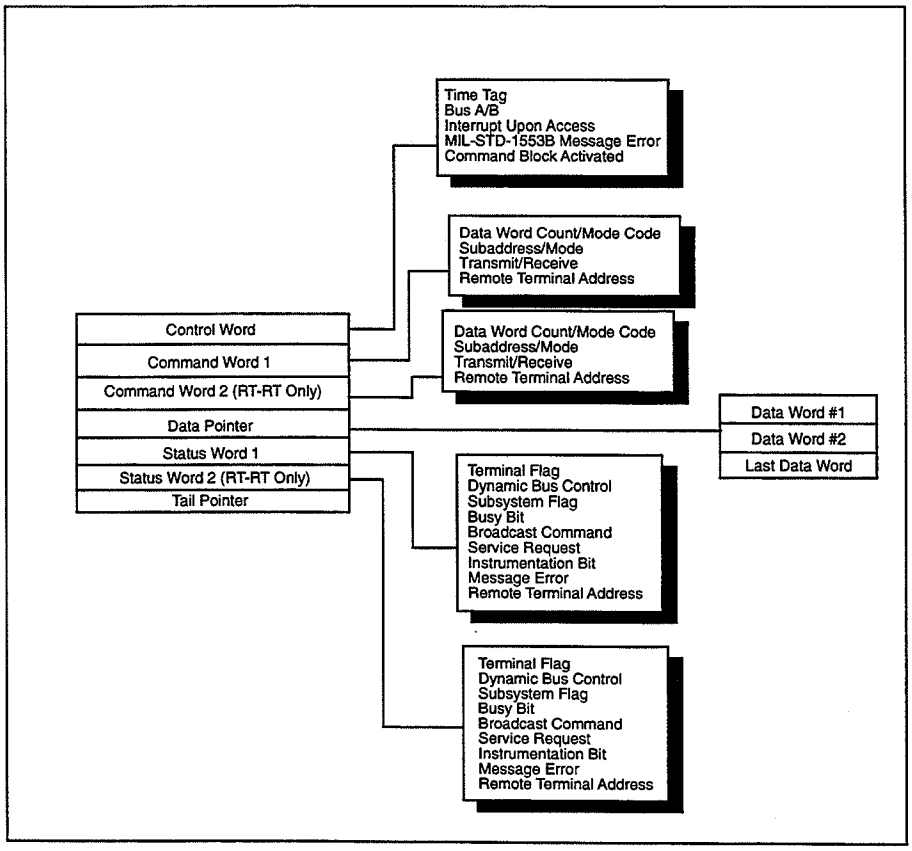


Figure 7-2 Monitor Terminal Structure

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

7.2.1 Control/Status Word

The user accesses the control/status word to determine which monitor terminal command block was last utilized and when particular RT activities occur (Figure 7-3).

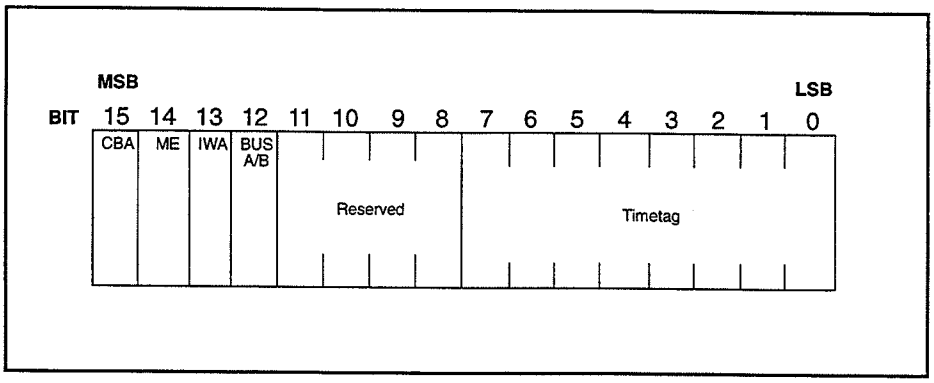


Figure 7-3 Monitor Terminal Control/Status Word

Table 7-1 describes the bits in the Monitor Terminal Control/Status Word.

Table 7-1 Monitor Terminal Control/Status Word

Bit	Function	Description
0-7	Time Tag	Bits 0-7 indicate the time in 64-microsecond resolution when a message was recorded. For example, if Bits 0-7 read 0000 0010, the message was completed 128 to 191 micro seconds after the timer was reset. The timer is reset via Register 13.
8-11	Reserved	
12	Bus Indicator (BA/B)	Bit 12 specifies whether the message was received on Bus A or B (1 = Bus A, 0 = Bus B).
13	Interrupt On Access (IOA)	If set, Bit 13 enables an Interrupt On Access to the particular monitor terminal command block. Bit 13 generates an interrupt when the block is accessed. Bit 13 can be used to generate an interrupt when the first message is received or to generate an interrupt with the occurrence of every n message ("n" is a multiple of the total number of monitor command blocks). If an interrupt is set every five messages, an interrupt is received at the end of the 5th, 10th, 15th, etc. block.
14	Message Error (ME)	If set, Bit 14 indicates the MIL-STD-1553B message contains an error.
15	Message Indicator	Bit 15 of the control word indicates whether data has been written to the Monitor terminal command block.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

7.2.2 Command Word One

Command Word One is used for all MIL-STD-1553B transfers and contains the RT address, subaddress and the word count or mode code. The first command of a Remote-to-Remote message type is a Receive command.

7.2.3 Command Word Two

Command Word Two is used when performing RT-to-RT transfers. It contains the second command for RT-to-RT transfers, which is the transmitting RT address, subaddress and the word count. Command Word Two is not used in single RT transfers.

7.2.4 Data Pointer

The Data Pointer is a 16-bit word that points to the starting location of the DTI-1120's Dual Port RAM, where the current message record block data buffer resides. The number of data words per buslist instruction equals 32 or less.

7.2.5 Status Word One

Status Word One contains one of the following:

- Remote terminal response status for BC-to-RT transfer (Master-to-Remote transfer) or RT-to-Bus Controller transfer (Remote-to-Master transfer)
- Status word of MIL-STD-1553B Mode Code transfer
- Remote terminal transmit status for RT-to-RT Transfer

7.2.6 Status Word Two

Status Word Two contains the receiving status in RT-to-RT transfers. If RT-to-RT transfers are not occurring, Word Two is not used.

7.2.7 Tail Pointer

The 16 bits of the tail pointer contain the starting address of the next sequential message record block.

7.3 Monitor Error Detection

In the Monitor Terminal Mode, the 1120 checks all monitored messages for errors. Detectable errors include Word Count, Long and Short Word, Manchester, parity and data contiguity errors.

Some types of errors are difficult for any monitoring device to interpret. For example, suppose an RT incorrectly responds to a Broadcast Command. The 1120, which is not receiving or transmitting the message cannot distinguish between the erroneous status word and a new command sent from the bus controller, since the Status Sync is identical to the Command Sync. In this case, the 1120 will place the extra status word in a new monitor terminal command block and report a Message Error caused by the incorrect protocol on the erroneously interpreted status word the RT transmitted.

7.4 Data Buffers

If memory is cleared, all data buffers contain zeros.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

7.5 Monitor Terminal Registers

Three registers are unique to the 1120: The Monitor Terminal Control Register (14) and the Monitor Terminal Address Select RT Registers (16 and 17). These registers control the 1120's Monitor Terminal Mode.

Other registers affecting the monitor function include: the Control Register (00), the Current MT Command Block, Current Buslist Instruction Address and Subaddress Response Block Address Register (02), the Interrupt Queue Pointer Register (06), the Standard and High Priority Interrupt Enable Registers (09 and 07), the RT Address Register (10) and the RT Timer Reset Register (13). (See Chapter 3 REGISTERS, for more information on Registers.)

7.6 Data Collection

To avoid overwriting data stored in memory, generate interrupts at message intervals and write the collected data to disk. Assume the MT contains 2000 MT command blocks with an interrupt generated every 500 blocks, meaning the interrupt bits of Blocks 500, 1000, 1500 and 2000 are set. If, at the interrupt, a message is stored in any of the blocks, the present block and the previous 499 blocks can be written to disk.

The monitor terminal command blocks and data buffers shown in Figure 7-4 maintain a one-to-one relationship. The order of the memory might be like the following:

Message record block 1	or	Monitor Block 1
Message record block 2		Data Buffer 1
Data Buffer 1		Message record block 2
Data Buffer 2		Data Buffer 2

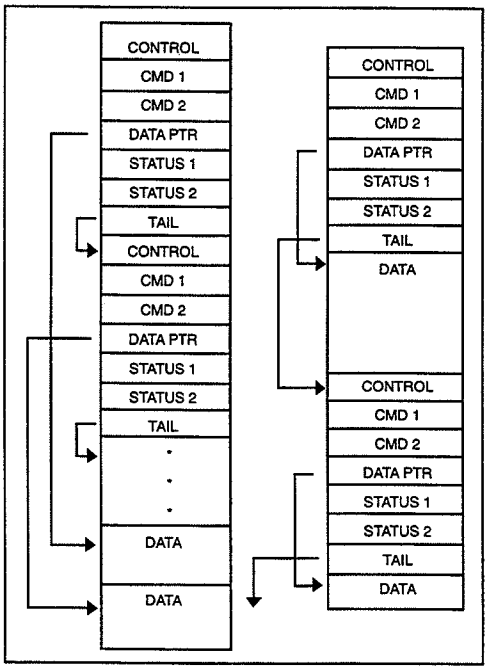


Figure 7-4 MT Command Blocks and Data Buffers

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

7.7 Post-Run Processing

Interrupts are generated so that the user can examine each message. The nature of each interrupt should also be examined to determine if an interrupt was expected and to determine why an unexpected interrupt may have occurred. An interrupt may occur if Bit 15 of the Control Word is set (indicating the block contains a message) or Bit 14 of the Control Word is set (indicating a message error has occurred).

7.8 Monitor Terminal Operational Sequence with Interrupts

1. Configure the monitor command in the preferred memory area.
2. Set up the data list pointers of the monitor command blocks to store all data words in the preferred memory area.
3. Verify the BC/RT Bit (Bit 10) in Register 0 reads 0. The RTs to be monitored can be enabled via Registers 16 and 17.
4. Ensure that the RT Address and Parity Bit in Register 10 is correct.
5. Enable the Standard Interrupt Bit in the High Priority Interrupt Enable Register. Also enable the Command Block Bit in Register 9 and the Command Block Access Bit in the control/status word of the monitor terminal command block.
6. Set the Enable Bit in the MT Control Register 14.
7. When message monitoring begins, the command, status and data words will be written into memory.
8. If an RT-to-BC message is being monitored, the command word is stored at CMD Word 1 of the message record block. The status is stored at Status Word 1 of the message record block and the incoming data words are stored in memory as pointed to by the data list pointer.
9. Having encountered the Command Block Access Bit, the 1120 asserts a Standard Interrupt and logs the event in the interrupt queue.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8.0 BOARD PROGRAMMING

8.1 Overview

After the DTI-112X is mapped but before it is operational, the 112X should be reset. To reset the 112X, READ Master Reset Register (18), READ any other register (e.g., Status Register), and WRITE to Master Reset Register. The user should load all memory locations with a value of zero.

The RESET/CLEAR operations are listed below:

- Reset the sequencers
- Initialize all buffers to zero
- Halt the DTI-112X operation if it is currently running
- Validate values in memory
- Clear the Control Register
- Clear the interrupt counter



NOTE: Memory is not affected by the reset operation.

8.2 Accessing the On-board Memory and Registers

8.2.1 Memory Access

The user can enable memory by setting Bit 0 of Register 18.

8.2.2 Creating the Interrupt Queue

An interrupt queue maintains a history of the generated standard interrupts. This queue can hold a maximum of 255 interrupts. Each interrupt block in the interrupt queue consists of three words. The third word in each block contains the offset from the beginning of the 112X memory to the first word of the next block (Figure 8-1).

During the initialization phase, these pointers can be stored in the 112X's on-board memory. It is recommended that users create contiguous interrupt blocks. The user can set the tail pointer of the interrupt queue to point to any desired entry.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

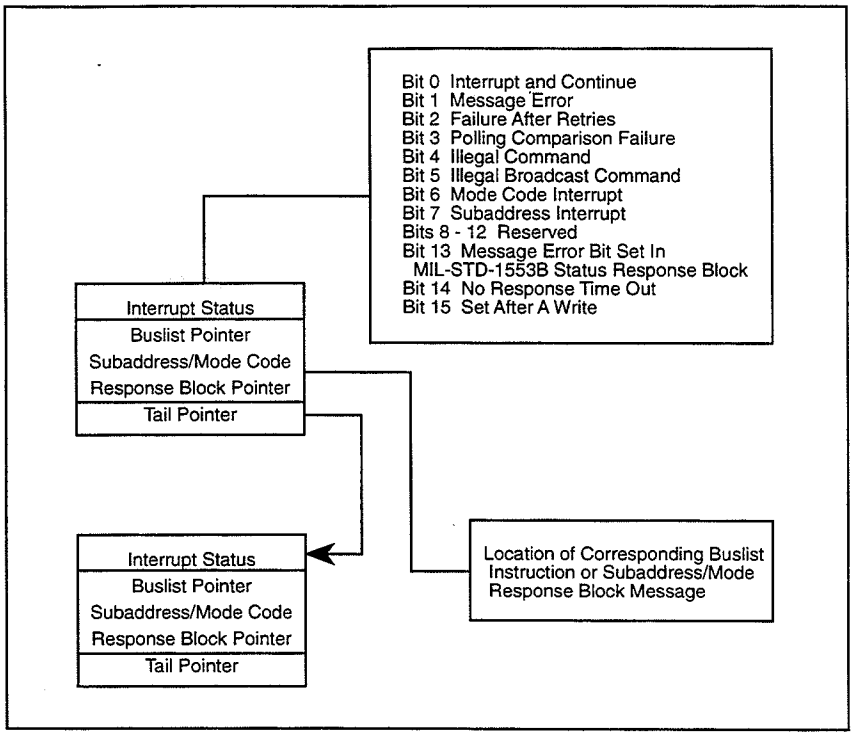


Figure 8-1 The Interrupt Queue

The circular queue maintains the most recent interrupt blocks, overwriting the oldest block when a new interrupt is logged. To set a circular queue, set the tail pointer of the last block to point to the first word of the first block. A queue can also be set so that the tail pointer of the last block points to itself.

Two registers—the Interrupt Counter, and the Autodecrement Interrupt—track the count of interrupt blocks in the interrupt queue. The Interrupt Counter Register returns the interrupt block count without affecting the number of interrupts counted. The Autodecrement Interrupt Counter Register automatically decrements the interrupt block count by one and returns the interrupt block count. If the interrupt queue is empty and the Autodecrement register is read, the count equals 255.



NOTE: To ensure accurate reporting of the interrupt block count, always read the Interrupt Counter Register.

To maintain an accurate correspondence between the interrupt count in the register and the number of interrupt blocks processed, read the Autodecrement Interrupt Counter Register as each interrupt block is processed.

Interrupt blocks are overwritten only if the interrupt count exceeds 255. When this occurs, the last interrupt block is overwritten, causing a loss of interrupt information during the simulation.

An Exceed Bit, returned with the interrupt count, informs the user that an overwrite has occurred. An overwrite in the Interrupt Counter Register is indicated by a returned value of 256 or greater in the low-order nine bits. Once set, this Exceed Bit remains set until the interrupt count is cleared.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

To clear the interrupt count and reset the Exceed Bit, write to the Interrupt Counter Register.

8.3 Bus Controller Mode

The Bus Controller Mode maintains the interrupt queue, buslists and data buffers (see Figure 8-2).

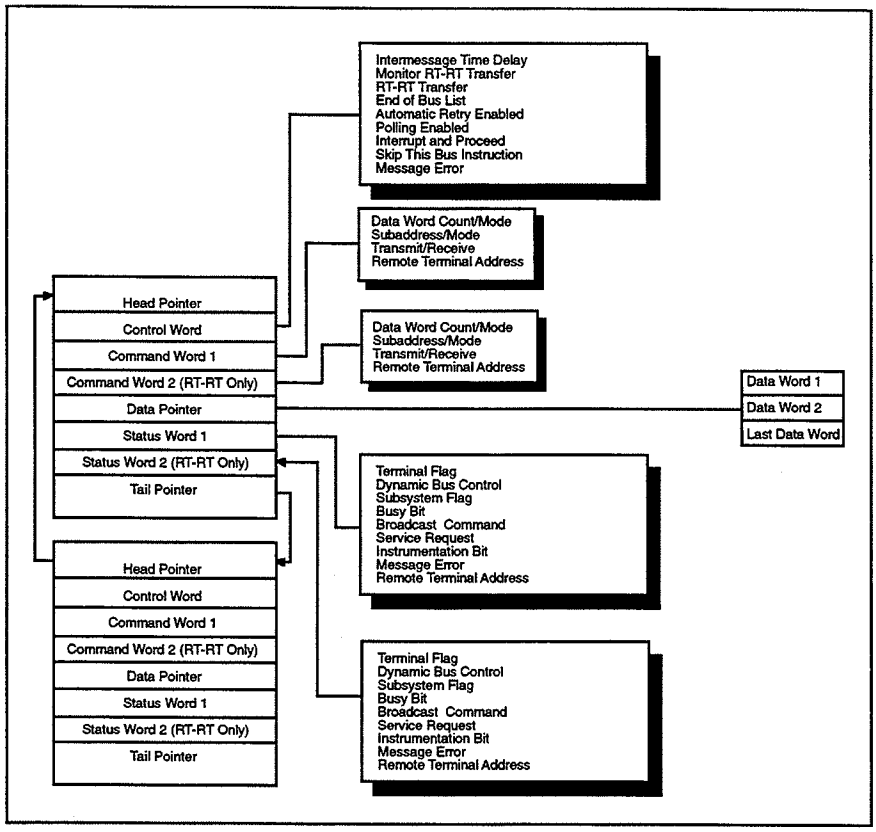


Figure 8-2 Bus Controller Structure

8.3.1 Creating a Buslist

Each buslist instruction block consists of eight words. Since each instruction block is linked to the previous and next instruction blocks via the head and the tail pointers, the instruction blocks do not need to be contiguous; however, if the instruction blocks are stored contiguously, traversing the buslist while viewing memory or memory dumps is easier.

The status words should be set to FFFF *hex* during loading of buslist instruction blocks. During post-run processing, the value of FFFF *hex* can be used to determine whether the remote terminals responded to the commands issued on the bus. A status response of FFFF should never appear on the bus. Buslist execution terminates when an instruction block with the End-Of- List Bit set in the control word is encountered.

It is best not to generate any MIL-STD-1553B traffic on the last command block. Set the Skip Bit and the End-Of-List Bit in the Control Word to ensure that the command words in the last instruction block are ignored.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

For repetitive execution of a one or more instruction blocks, set the tail pointer of the last instruction block in the sequence to point to the first word of the first instruction block in the sequence. Buslist execution is terminated by setting the End-Of-List Bit of an instruction block control word in the loop while the buslist is executing. Buslist execution can also be terminated with a board reset by accessing the Reset Command Register or the Master Reset Register.

Each instruction block contains one data pointer. The data pointers do not need to be unique for each instruction block. The same buffer of data can be pointed to by the data pointers for multiple instruction blocks. In the looping example of the preceding paragraph one data pointer exists per instruction block . The transmitted data is re-transmitted and the received data is overwritten each time the instruction block is executed.

To change one or more of the data pointers each time the buslist is executed, set the control word interrupt bit of the last instruction block, which may contain an intermessage delay. The intermessage delay value should be long enough to allow detection of the interrupt and modification of the data pointers. When the delay ends and the buslist continues, the data pointed to by the modified data pointers is transacted.

Multiple buslists can be created and stored in memory. Each buslist is executed by loading the address of the first instruction block of a particular buslist into the Current Buslist Instruction Register and by loading the bit pattern of the desired bus controller options with Bit 0 set in Register 0.

If the High Priority Message Error Interrupt is enabled and a message error occurs, the bus controller halts execution until the interrupt is cleared by writing a value of '2' to the High Priority Interrupt Status/Reset Register. Writing a high signal to a bit in the High Priority Interrupt Status/Reset Register resets the bit.

When the Message Error Interrupt Bit is reset in the High Priority Interrupt Status/Reset Register, the Control Word Message Error Bit of the buslist instruction block is set and the bus controller resumes buslist execution. The Message Error Bit remains set even if the instruction is re-executed and a message error does not occur.

8.3.2 Initializing Data to be Transacted

When memory is cleared, all Receive data buffers contain zeroes. Transmit data buffers must be loaded with the appropriate data, which is the first step in the initialization process. The data pointer in the transmitting buslist instruction block points to the first data value to be stored or transmitted. The number of data words to be transmitted is contained in the word count field of the transmitting instruction block command word.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8.3.3 Loading Bus Controller Registers

The next step in the initialization process is to load the bus controller registers.

Table 8-1 Loading the Bus Controller Registers

Controller Register	Loading the Register
Interrupt Queue Pointer Register (06)	To generate interrupt blocks, write the pointer at the top of the interrupt queue to Register 06.
Polling Compare Register (03)	If enabling polling, load the Polling Compare Register with the desired polling compare value.
Current Buslist Instruction Register (02)	Write the address of the beginning of the buslist to Register 02.
High Priority Interrupt Enable Register (07) and Standard Interrupt Enable Register (09)	Interrupts are enabled/disabled by writing to Registers 07 and 09. To determine when the buslist has completed execution, enable the End-Of-List Interrupt and wait for the interrupt to be generated. Include a time-out loop in case an unpredictable error occurs and prevents the End-Of-List Interrupt from being generated.
Control Register (00)	The Bus Controller options can be stored in Register 00 at the same time the Start Bit (Bit 0) is set. Bit 10 of Register 00 must be set for the 112X to operate as a Bus Controller. Buslist execution begins as soon as Bit 0 is set.

8.3.4 Post-Run Processing

Examine the results of the buslist execution by viewing the High Priority Interrupt Status/Reset Register (Register 8) and the Interrupt Queue (if Standard Priority Interrupts were enabled). Check the MIL-STD-1553B status words stored in the buslist instruction blocks, the control word Message Error Bit of the instruction blocks and all received data.

To determine which of the enabled High Priority Interrupts have occurred, examine the corresponding interrupt bits set in the High Priority Interrupt Status/Reset Register (Register 8). If Standard Interrupts were enabled and at least one Standard Interrupt was generated, the least significant bit of Register 8 is set. Ascertain the cause of the Standard Interrupt by viewing the interrupt blocks on the interrupt queue. Verify that only expected interrupts were placed on the interrupt queue.

The MIL-STD-1553B status words resulting from the buslist instruction block execution are stored in the same instruction block. For example, if the status words in the instruction block were initialized to FFFF *hex* and the first status word for a Non-Broadcast transaction is FFFF, the remote terminal failed to respond.

Similarly, if the second status word for a Non-Broadcast Remote-To-Remote transaction is FFFF, the remote terminal receiving the Remote-To-Remote transfer did not respond. If the remote terminal address portion of the status words are correct but the low-order eleven bits of the status words are non-zero, a status exception occurred.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

If a Message Error Interrupt is not enabled, a message error occurring during the buslist instruction block execution will still cause the most significant bit of the instruction block control word to be set. The Message Error Bit of all instruction blocks should be scanned to determine if a message error occurred during instruction block execution. Use the corresponding data pointer in each instruction block to locate and verify the data words received by the bus controller.

8.4 Remote Terminal Mode

The Remote Terminal Mode maintains the interrupt queue, subaddress/mode code response blocks, data buffers and message status words (Figure 8-3). Refer to the Interrupt Queue section of this chapter for specific interrupt information. The following sections describe subaddress/mode code response blocks, data buffer initialization, register loading with RT options and post-run processing.

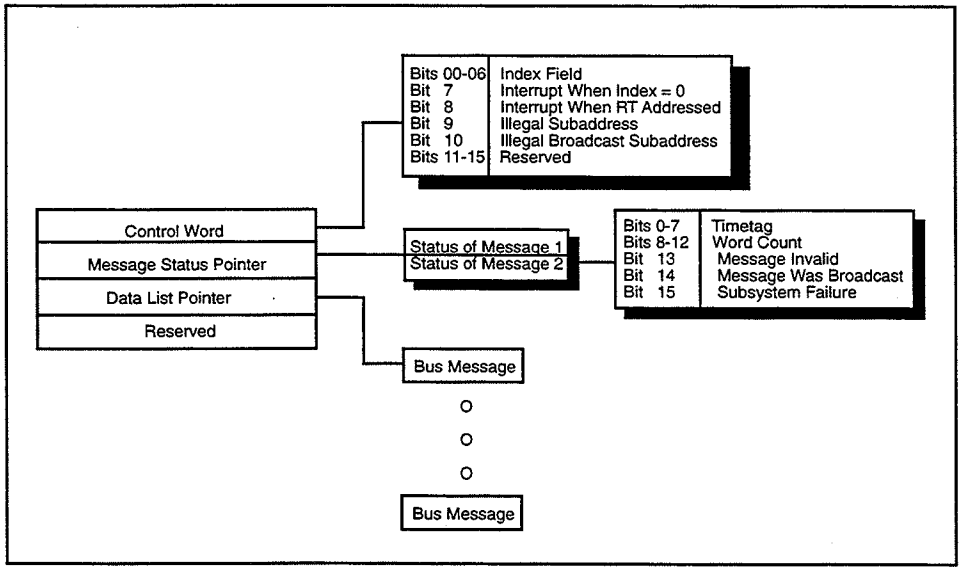


Figure 8-3 Remote Terminal Structure

8.4.1 Subaddress/Mode Code Response Blocks

The subaddress/mode code response blocks require 320 words of memory and must begin at an address that is evenly divisible by 512 (200 *hex*). Each response block is stored in a contiguous manner and consists of four words. Table 8-2 shows the function of these response blocks. Table 8-2 Function of Response Blocks

Response Block	Function
First 30 (or 31)* blocks	Contain information about Receive Subaddresses 1-31
Next block	Reserved
Next 31 blocks	Provide information about Transmit Subaddresses 1-31
Next block	Reserved
Next 16 blocks	Contain information about mode codes (two mode codes, one without data and one with data, per block)

*Depending on whether Subaddress 31 is configured as a data or mode subaddress

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

INITIALIZATION

The following options apply to initializing the subaddress/mode code response blocks:

- All subaddresses and mode codes can be DISABLED except for selected subaddresses and mode codes of interest, which are ENABLED.
- All Subaddresses and mode codes can be ENABLED.

Usually, either of these options can be selected.



NOTE: On a 32 K board, the maximum number of buffers allotted per subaddress/mode code is 128. The maximum number of data words possible, per MIL-STD-1553B transmission, is 32. This maximum limit (32) cannot be reserved for every subaddress/mode code. Thus, the application requirements may dictate the selection of a particular option. Do not exceed memory capacity.

The mode code responses should be initialized per MIL-STD-1553B.

CONTROL WORD

The control word of each subaddress/mode code response block contains the following interrupt bits:

- Bit 8 — Interrupt When Addressed
- Bit 7 — Interrupt When Index = 0 (when overwriting begins)

When set, these bits enable certain Standard Interrupts.

When Standard Priority Interrupts are enabled by setting the least significant bit of Register 7, addressing a subaddress/mode code with the **Interrupt When Addressed** bit set generates an interrupt.

The **Interrupt When Index = 0 Bit** provides a means to generate an interrupt when all subaddress/mode code buffers have been transmitted or received. When the index is decremented from '1' to '0' and Standard Interrupts are enabled, an interrupt is generated. No additional interrupts result from the event if the subaddress/mode code is involved in more transactions.

Since a mode code without data and a mode code with data both use the same response block, two types of interrupts are used to differentiate them. A mode code without data generates a specific **Mode Code Interrupt** (Bit 6 of the Interrupt Status Word), while a mode code with data generates a **Subaddress Event Interrupt** (Bit 7 of the Interrupt Status Word).

MESSAGE STATUS AND DATA LIST POINTERS

Pointers to a message status list and a data buffer list are associated with each subaddress/mode code block. The values of these pointers depend on the user application.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

For example, if a single data buffer of 32 words is reserved for each subaddress, the first buffer's address can be stored in the response block for the first subaddress. The data list pointer for succeeding subaddresses is calculated by adding 32 to the data list pointer for the previous subaddress.

Only one data word can be transacted with a mode command. A data buffer of one word needs to be reserved for each mode code with data in a single buffer scheme.

Only one message status word is required per subaddress in a single buffer situation. The message status word for one subaddress/mode code is stored in the location following the message status word of the previous subaddress/mode code.

In a multi-buffering situation, the data involved in the transactions for the subaddress are stored contiguously. Two examples follow to show how data is transmitted and received in multi-buffering situations.



NOTE: If the index is initially set to a value greater than zero and enough messages are transacted so that the index field changes from one to zero, the data list and message status pointers indicate the next buffer to be accessed.

When the next message to the subaddress/mode code is transacted, the index remains zero, the pointers point to the last message transacted and no data is overwritten.

When additional messages to the subaddress/mode code are transacted, the index remains zero and information about only the last message can be examined. It is the user's responsibility to set the index field to a large enough value (up to 128) to ensure data is not lost.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

EXAMPLE 1 — TRANSMITTING DATA

IF

- The index field in the control word of a transmit subaddress response block is set to '1'
- The data list pointer for this subaddress points to the sequence of values 1, 2, 3, 4, 5, 6 and 7
- The subaddress receives a command to transmit three words

THEN

- The values 1, 2 and 3 are transmitted
- The data list pointer now points to the location containing the value 4

The location pointed to by the message status pointer for this subaddress is modified to contain a word count of 3.

The pointer points to the next memory location, and the index is decremented so it now contains zero.

If the **Interrupt When Index = 0** Bit is set in the control word of the response block, an interrupt block is added to the interrupt queue. If the subaddress is requested to transmit four more words, the values of 4, 5, 6 and 7 are transmitted. Since the index equals zero, the data list pointer is not updated.

Because the index field was already '0', no interrupt block is added to the interrupt queue. The location pointed to by the message status pointer is loaded with the word count of 4 and the message status pointer is not incremented.

IF

- The subaddress is requested to transmit two words

THEN

- The values of 4 and 5 are transmitted
- The word count field of the last message status word is overwritten with a value of 2
- Neither pointer is updated

IF

- The subaddress receives a command to transmit five words

THEN

- The value 4, 5, 6, 7 and the value in the next location are transmitted
- The word count field of the last message status word is overwritten with a value of 5
- The pointers are unchanged

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

EXAMPLE 2 — RECEIVING DATA

IF

- The index field in the control word of a receive subaddress response block is set to '1'
- The subaddress receives a command to receive three words

THEN

- The three received values are stored in three consecutive locations starting at the location indicated by the data list pointer
- The data list pointer now points to the location following the last data word received.

The message status for this subaddress is modified to contain a word count of 3, the message status pointer points to the next memory location, and the index is decremented so it now contains a zero.

If a message error is detected, the **Message Error** Bit in the message status word is set. The message status word pointer is incremented, and the index is decremented. However, the data list pointer is not changed except for word counts that are high or low. In the case of high or low word counts, the data list pointer is incremented by the number of data words specified by the MIL-STD-1553B command word.

IF

- The Interrupt When Index = 0 Bit is set in the control word of the response block

THEN

- An interrupt block is added to the interrupt queue.

IF

- This subaddress is requested to receive four more words

THEN

- The data words are stored in four consecutive locations beginning with the location specified by the data list pointer.

The data list pointer is not updated, since the index is zero.

Since the index field was already 0, no interrupt block is added to the interrupt queue. The location pointed to by the message status pointer is loaded with the word count of 4, and the message status pointer is not incremented.

(Continued on next page)

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

EXAMPLE 2 — RECEIVING DATA (continued from previous page)

IF

- The subaddress is requested to receive five words

THEN

- The data values overwrite the last four data values received plus the next memory location
- The word count fields of the last message status word are overwritten with the value 5
- Neither pointer is updated

IF

- The subaddress receives a command to receive two words

THEN

- The data words overwrite the first two words of the last message but the other three words from the last message are not altered
- The word count field of the last message status word is overwritten with the value 2
- The pointers are unchanged

The data list pointer for Mode Codes 18 (Transmit Bit Word) and 19 (Transmit Last Command) are not used to retrieve the BIT word and last command. The BIT word is retrieved from the Bit Word Register. The word count field in the message status word indicated by either of these mode codes contains a '0' instead of '1' after the mode codes are transacted.

If the High Priority Message Error Interrupt is enabled via the High Priority Interrupt Enable Register and a message error occurs, remote terminal operations stop until cleared.

The interrupt is cleared by writing the Value '2' to the High Priority Interrupt Status/Reset Register. Writing a high signal to a bit resets it.

When a value of '2' is written to the High Priority Interrupt Status/Reset Register, the message error bit in the message status word of the subaddress/mode code response block is set and the remote terminal resumes operation.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8.4.2 Initializing Data to be Transacted

If memory has been cleared, all Receive data buffers contain zero values.

8.4.3 Loading RT Registers

The next step in the initialization process is to load the remote terminal registers.

INTERRUPT QUEUE POINTER REGISTER

If interrupt blocks will be generated, the address of the beginning of the interrupt queue is written to the Interrupt Queue Pointer Register.

TRANSCEIVER BUILT-IN TEST WORD REGISTER

The Built-In Test word is written to this register.

REMOTE TERMINAL TIMER RESET COMMAND REGISTER

Writing any value to this register will reset the RT Timer. The RT timer is the source of timetag information in the message status words.

REMOTE TERMINAL ADDRESS REGISTER

The Remote Terminal address is written to this register along with RT options. Bit 5 of this register must also be set if the RT address contains an even number of bits.

CONTROL REGISTER

Load a value into the Control register to enable either or both buses and to start RT operation.

8.4.4 Post-Run Processing

The validity and accuracy of transacted data are determined by viewing the High Priority Interrupt Status Register, the Interrupt Queue (if interrupts were enabled), the message status words for all messages and all transacted data.

The nature of any High Priority Interrupts that occurred can be determined by examining which bits are set in the High Priority Interrupt Status/Reset Register.

If standard interrupts were enabled and at least one standard interrupt was generated, the least significant bit of the High Priority Interrupt Status/Reset Register is set. The cause of standard interrupts can be ascertained by viewing the interrupt blocks on the interrupt queue. Verify that only expected interrupts were placed on the interrupt queue. The verification process can be combined with data validation (described later in this section).

There are several ways to examine data transacted during the scenario depending on the user's application. One way to examine transacted data is to examine each subaddress/mode code in succession. If the index field of a receive subaddress/mode code contains a zero, the transacted messages may have been overwritten.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

To ascertain how many messages were transacted and lost, set the **Interrupt When Addressed** Bit in the control word of each subaddress/mode code response block the next time this scenario is executed

If the interrupt queue is large enough to contain all the interrupt blocks that are generated, this queue also provides a count of the times each subaddress/mode code was addressed and a chronological order of bus activity.



NOTE: The message status and data list pointers point to information concerning the most recent message.

IF

The subaddress/mode code transacted multiple messages and the index field was initially non zero

THEN

It is necessary to return to the locations initially indicated by the data list and the message status pointers.

There are several ways to return to the indicated locations.

1. The host software could maintain a list of the indicated locations.
2. The on-board memory could also be initialized and a special value (such as FFFF *hex*) could be placed at the beginning of the data list and message status list. The pointer indicates the location after the special value.
3. When simulation is over, the message status pointer could be loaded into a variable and the variable's value could be decremented so that it points to the message status word of the previous message. The word count field is extracted from the indicated message status word.
4. The data list pointer could be loaded into a variable and the variable's value could be decremented by the word count so the variable points to the first word of the message. Repeat the process until the special value is encountered. The first buffer can now be examined.

The method used to process data depends on many factors:

- The number of messages transacted
- The number transacted per subaddress/mode code
- The number of subaddress/mode codes enabled

Regardless of how messages are examined, the status word indicates:

- A message error
- A subsystem failure
- The number of data words transacted

If the message is valid, view the number of data words specified by the word count and verify their accuracy.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8.5 Bus Monitor Mode

The Bus Monitor Mode maintains the following:

- Interrupt queue
- Monitor command blocks
- Data buffers

This section describes message record block initialization, bus monitor register loading, and data processing.

8.5.1 Initializing the Monitor Command Blocks

Each message record block contains seven words, previously defined in this manual. Because each message record block is linked to the next block through the tail pointer, the message record blocks do not need to be contiguous. It may be helpful to store the data words immediately after the associated message record block.

The tail pointer of the last block is typically set to point to itself or to the first message record block. If the tail pointer points to itself, only the last block is overwritten when another message is captured.

EXAMPLE

If "n" monitor command blocks were created, the first "n-1" messages that were stored are saved along with the most recent message.

If the tail pointer points to the first message record block, the oldest message is overwritten when a new message is captured; therefore, the most recent "n" messages are maintained in the monitor command blocks.

If desired, the status words could be loaded with a value (such as FFFF *hex*) that never appears on the MIL-STD-1553B bus. Then when examining the status words in the monitor command blocks after traffic is recorded, any status words that appear on the MIL-STD-1553B bus are obvious.



NOTE: The algorithms presented in this manual do not load the status words with a special value before data collection begins.

Once running, the bus monitor can be stopped only by writing a value (with the least significant bit clear) to Register 0, or by resetting the board (by accessing Register 12 or 18).

Each message record block contains a data pointer. To ensure 1553B data words are stored properly, set the data pointer for each block before the bus monitor begins operation. Since a maximum of 32 data words can be specified in a command, storage for 32 words should be reserved for each message. Part of the on-board memory is not used if the message contains less than 32 data words.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

If the number of transmitted data words exceeds the number of data words specified by the word count field of the recorded command, the bus monitor stores the number of data words specified by the word count. Extra data words are discarded.

The **Interrupt When Accessed** Bit in the message record block control word can be used to provide important information to the application program. If the Interrupt Bit is set in the first message record block, the software is notified when data recording begins. If the Interrupt Bit is set in every 200th block, the software can store the 200 messages to disk for later processing.

Traffic can be monitored on both buses or either the primary bus or secondary bus. Two other recording options are available. Either all traffic that appears on the enabled buses is recorded or only the traffic involving remote terminals enabled for monitoring will be captured.



NOTE: If only one remote terminal in a remote-to-remote command is enabled for monitoring, the amount of traffic captured depends on the command in which the RT address appears. If the RT address appears in the receive command (first command), the entire message is captured. If the RT address appears in the transmit command (second command), the receive command and receive status are not recorded

If the High-Priority Message Error Interrupt is enabled and a message error occurs, the Bus Monitor stops executing until the interrupt is cleared by writing the value '2' to Register 8. Writing a high signal resets the bit. When a value of '2' is written to the Interrupt Status/Reset Register, the **Message Error** Bit in the control word of the message record block is set and the bus monitor resumes activity.

8.5.2 Initializing Data Buffers

Since memory was cleared, all buffers contain zeros.

8.5.3 Opening a Disk File

If data is to be stored on disk, the disk file should be opened before data collection begins.

8.5.4 Loading Bus Monitor Registers

The registers can now be loaded. Table 8-3 shows the contents of these registers:

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Table 8-3 Load Bus Monitor Registers

Register Number	Contents
0	Value that enables buses and begins monitoring operations
2	Location of the first message record block
6	Pointer to the top of the interrupt queue (if interrupt blocks are to be generated)
7 & 9	Interrupt Enable
14	Value that places the board in Bus Monitor Mode and selects all or RT Recording Mode
16	Values for selecting remote terminals (if traffic to specified remote terminals is recorded)
17	Values for selecting remote terminals (if traffic to specified remote terminals is recorded)



NOTE: Bit 10 of Register 0 must be clear when functioning as a bus monitor.

8.5.5 Saving Data to Disk

If enough bus traffic is monitored, all monitor command blocks will eventually contain messages. To prevent data from being overwritten, periodically write captured traffic to a disk. An interrupt can be generated to notify the user that a specified number of monitor command blocks are filled and data should be saved to a disk. After writing to a disk, eliminate any leftover values that may interfere with the analysis of the next message that is written to the block by writing a '0' to the control word, to both command words, and to both status words.

8.5.6 Post-Run Processing

When monitoring activity halts, post-run analysis can begin. If data was written to disk, save the current buffer if it contains any data. Close the disk file. The disk file is re-opened and records are analyzed until the end-of-file is detected. If data was not saved on disk and only information in on-board memory is examined, the first record to be examined is pointed to by the original value of Register 2. Analysis ends when a block is encountered with Bit 15 of the control word reset (not all blocks were used), or when the address of the next block equals the value stored in Register 2 (all blocks were filled and have been examined).

FIRST WORD OF MESSAGE RECORD BLOCK

Retrieve the first word of the message record block, the control word, under examination.

If the most significant bit is not set, the message record block does not contain any data and post-run processing is finished. Check Bit 14 . If Bit 14 is set, a message error was detected during the transaction. Examine Bit 12 to determine which bus the message was

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

transacted on (set if Bus A, clear if Bus B). The least significant eight bits of the control word contain the timetag (in 64 microsecond increments).

SECOND WORD OF MESSAGE RECORD BLOCK

Obtain the second word of the message record block, which is the first command word.

The RT address appears in the most significant five bits of the first command word. The direction of the command is encoded in the next bit (set if transmit or clear if receive). The next five bits contain the subaddress. If the subaddress is a mode subaddress, the last five bits represent the mode code. Mode codes greater than 15 involve a data word. Mode codes less than 16 do not involve a data word.

THIRD WORD OF MESSAGE RECORD BLOCK

If the message was a remote-to-remote transfer, the next word contains the second command. The second command is a transmit command, meaning the Direction Bit in the command word must be set and the second command word is non-zero. The RT address and subaddress field can be examined with the same technique that was used on the first command word. The word count field should be identical to the word count field in the first command.

FOURTH WORD OF MESSAGE RECORD BLOCK

The fourth word in the message record block points to the data words included in the message. If the command was a mode code without data, no data words exist to examine. Otherwise, validate the number of data words contained in the message.

FIFTH WORD OF MESSAGE RECORD BLOCK

The next word is the first status word. If the status words were initialized to FFFF *hex* and the current value is still FFFF, the status word did not appear on the bus. If the value of the first status word is not FFFF, examine the low order (11) bits for a status exception.

SIXTH WORD OF MESSAGE RECORD BLOCK

If the message was a remote-to-remote transfer, the sixth word in the block is the second status word. If the status words were initialized to FFFF and the current value is still FFFF, the status word did not appear on the bus. If the word's value does not equal FFFF, examine the low order (11) bits for a status exception.

SEVENTH WORD OF MESSAGE RECORD BLOCK

The last word of the block points to the next message record block. Continue to analyze each block until all blocks have been examined.



NOTE: If a data file was open for analysis, close the file before exiting the program.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8.6 Algorithms

At least two examples of setting the 112X in each operating mode are provided. The first example of each set creates a simple environment that shows recommended initialization procedures and the minimum requirements to run in the selected operational mode. The other examples expand this environment to include more options available in the selected mode of operation. The notation "0x" means the succeeding characters are hex digits.

8.6.1 Bus Controller Example 1

The first bus controller example executes a buslist containing a single instruction block. No interrupts are generated . Upon completion of the buslist, the MIL-STD-1553B status word is checked for status exceptions and the message error bit of the control word of the buslist instruction block is examined to see if it is reset as expected.

BUS CONTROLLER EXAMPLE 1

```
/* define constants*/

mem_size = 16k or 32k, depending on switch setting
data_ptr = 0x1000
start_buslist = 0x0300
start_int_que = 0
/* base on-board memory and registers */

     base on-board memory
     if (base fails) terminate

     base on-board registers
     if (base fails) terminate

/* reset the board by reading register 18 (allow time for operation to complete) */

     dummy = dtireg(18)
     dummy = dtireg (1)

/* enable 112X memory */

     dtireg (18) = 1

/* clear on-board memory (mem_size is memory size in words) */

     do i = 0, mem_size-1
     dtimem(i) = 0
     enddo

/* create one instruction buslist */

     i = start_buslst
     dtimem(i) = -1                    /* head pointer can be any value */
                                       /* since there is no previous */
                                       /* instruction block */

     dtimem(i+1) = 0x400               /* set end-of-list bit */
     dtimem(i+2) = 0x0832              /* MR, RT1 SA1 WC18 */
     dtimem(i+4) = data_ptr            /* pointer to data to transmit */
     dtimem (i+5) = 0xFFFF             /* initialize status word to -1 */
     dtimem(i+7) = -1                  /* tail pointer can be any value since*/
                                       /* there is no next instruction */
                                       /* block. load arbitrary data values*/
                                       /* to transmit */
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Controller Example-1 is continued from the previous page.

```
     wrd_cnt = dtimem(i+2) AND 0x1F
     if (wrd cnt = 0) wrd_cnt = 32

     do j = 1, wrd_cnt
          dtimem(data_ptr+j-1) = j
     enddo

/* load registers */

     dtireg(7) = 0                        /* disable all interrupts */
     dtireg(9) = 0
     dtireg(2) = start_buslst             /* define start of buslist*/
     dtireg(0) = 0x0481                   /* enable bus a and start BC */

/* allow time for buslist to execute */
/* examine status words to verify traffic was generated */

     cntl = dtimem(i + 1)
     stat1 = dtimem(i + 5)

/* status should not be FFFF because initial value should be
   overwritten, message error bit should not be set in control
   word and status word should only contain remote terminal
   address */

     if (stat1 = 0xFFFF)
          display no status message
     else if (cntl AND 0x8000) ! = 0
          display message error
     else if (stat1 ! = 0x800)
          display status exception
     else

/* if this buslist contained commands that received data, this is
   the place where the data would be displayed */
     endif
```

8.6.2 Bus Controller Example 2

This algorithm describes the steps required to execute a multi-instruction block buslist that will generate standard and high priority interrupts. All the options of the buslist instruction block control word are used. Upon completion of the buslist, the same status word and control word checks that were used in the previous example are performed along with verification of expected interrupts and detection of unexpected interrupt blocks.

BUS CONTROLLER EXAMPLE-2

```
/* define constants */

     mem_size = 16k or 32k, depending on switch setting
     start_buslst = 0x0300
     num_instr = 8
     start_data = 0x1000
     start_int_que = 0
     num_blks = 255
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Controller Example-2 is continued from the previous page.

```
/* base on-board memory and registers */
     base on-board registers
     if (base fails) terminate

/* reset the board by reading register 18 (allow time for
   operation to complete) */

     dummy = dtireg(18)
     dummy = dtireg(1)

/* Enable 112X memory */

     dtireg (18) = 1

/* clear on-board memory (mem_size is memory size in words) */

     do i = 0, mem_size-1
          dtimem(i) = 0
     enddo

/* assuming interrupt blocks are contiguous, create interrupt
   block links for interrupt queue (remember that link of first
   block is two words after first word of that block) and let
   link of last block point to first block */

     do i = 1, num_blks-1
          dtimem(start_int_que + i*3-1) = start_int_que + i * 3
     enddo

     dtimem(start_int_que + num_blks *3-1) = start_int_que

/* create control word and command word(s) for each instruction
   block */

     cntl(1) = 0x1000                    /* polling enable */
     cntl(2) = 0x0300                    /* RT-RT cmd, monitor data */
     cntl(3) = 0x2000                    /* interrupt and proceed */
     cntl(4) = 0x6000                    /* interrupt and skip instruction block */
     cntl(5) = 0x0200                    /* RT-RT cmd, do not monitor data */
     cntl(6) = 0x0020                    /* delay 32 units (512 usec) */
     cntl(7) = 0x0800                    /* retry enable */
     cntl(8) = 0x0400                    /* end of list */
     cmd1(1) = 0x0823                    /* BC --> RT1 SA1 WC3 */
     cmd2(2) = 0
     cmd1(2) = 0x0823                    /* RR, RT2 SA2 --> RT1 SA1 WC3 */
     cmd2(2) = 0x1443
     cmd1(3) = 0x0c25                    /* RM, RT1 SA1 WC5 */
     cmd2(3) = 0
     cmd1(4) = 0x0c25                    / *RM, RT1 SA1 WC5 */
     cmd2(4) = 0
     cmd1(5) = 0x082a                    /* RR, RT2 SA2 --> RT1 SA1 WC10 */
     cmd2(5) = 0x144a
     cmd1(6) = 0x0c20                    /* RM, RT1 SA1 WC32 */
     cmd2(6) = 0
     cmd1(7) = 0x0841                    /* MR, RT1 SA2 WC1 */
     cmd2(7) = 0
     cmd1(8) = 0x0811                    /* MC, RT1 SA0 MC17 */
     cmd2(8) = 0

/* load head pointer, tail pointer, and data buffer pointer for
   each instruction block in buslist (the first head pointer and
   last tail pointer can be any value, 32 words reserved for each
   data buffer) */
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Controller Example-2 is continued from the previous page.

```
     do i = 1, num_instr
          addr = start_buslst + (i-1) * 8
          dtimem(addr) = addr - 8        /* head pointer */
          dtimem(addr + 1) = cntl(i)     /* control word */
          dtimem(addr + 2) = cmd1(i)     /* first command */
          dtimem(addr + 3) = cmd2(i)     /* second command */
          dtimem(addr + 4) = start_data + (i-1)*32
                    /* 32 words/buffer */
          dtimem(addr + 5) = 0xFFFF      /* status word 1 */
          dtimem(addr + 6) = 0xFFFF      /* status word 2 */
          dtimem(addr + 7) = addr + 8    /* tail pointer */
     enddo

/* load data values (assumed to be stored in array called
   "data_array" dimensioned (32,num)) */

     do i = 1, num_instr
          wrd_cnt = dtimem(start_buslst + (i-1)*8 + 2) AND 0x1F
          if (wd_cnt = 0) wrd_cnt = 32
               do j = 1, wrd_cnt
                    dtimem(dtimem(start_buslst + (i-1)*8 + 4) + j-1)
                    = data_array(j,i)
               enddo
     enddo

/* load registers */

     dtireg(3)= 0x400                    /* when polling, check for msgerr bit */
     dtireg(19) = 0                      /* clear interrupt block counter */
     dtireg(6)=start_int_que             /* definestartof interrupt queue*/
     dtireg(7) = 0x01A5                  /* enable certain high-pri interrupts */
     dtireg(9) = 0x000f                  /* enable certain standard
             interrupts */
     dtireg(2) = start_buslst            /* define start of buslist */
     dtireg(0) = 0x0483                  /* define retry, xmt on a, and
             start */

/* wait for end-of-list interrupt to occur, indicating that bus
   list has finished executing */

     do while (dtireg(8) AND 4) = 0)
     end while

/* examine status words to verify traffic was generated, verify
   ME bit not set in control word of buslist instruction block,
   and examine received data */

     do n = 1, num_instr

          i = start_buslst + (n-1) *8    /* address of instr block */
          cntl = dtimem(i+1)             /* control word */
          cmd1 = dtimem(i+2)             /* first command */
          cmd2 = dtimem(i+3)             /* second command */

          data_ptr = dtimem(i+4)         /* pointer to data words */
          stat1 = dtimem(i+5)            /* first status word */
          stat2 = dtimem(i+6)            /* second status word */
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Controller Example-2 is continued from the previous page.

```
/* ignore any instruction blocks with skip bit set */

          if ((cntl AND 0x4000) = 0)
/* check both status words of RT-RT for validity */

               if ((cntl AND 0x0200)! = 0)

/* no status response */

                    if ((stat1 = 0xffff) OR (stat2 = 0xffff))
                         display no status response message

/* status exception */

                    else if (((stat1 AND 0x07ff) != 0) OR
                         ((stat2 AND 0x07ff) != 0))
                         display status exception message

/* verify rt number - if remote-to-remote, first status is
   associated with second command and second status is associated
   with first command */

                    else if ((stat1 AND 0xF800) != (cmd2 AND 0xF800))
                         display incorrect rt address in status
                         if ((stat2 AND 0xF800) != (cmd1 AND 0xF800))
                              display incorrect rt address in status
                    endif

                    else

/* no status response */

                    if (stat1 = 0xFFFF)
                         display no status response message

/* status exception */

                         else if ((stat1 AND 0x07FF!= 0)
                              display status exception

/* verify rt number */

                         else if ((stat1 AND 0xF800) != (cmd1 AND 0xF800))
                              display incorrect rt address in status
                         endif
                    endif

/* check if message error bit is in control word */

                    if ((cntl_wrd AND 0x8000) !=0)
                         display message error occurred

/* validate received data */

                    endif

     enddo

/* check for unexpected high-priority interrupts (expect end-of-
   list and command execution (standard) interrupts) */
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Controller Example-2 is continued from the previous page.

```
     if ((dtireg(8) AND 0x01A5) !=5)
          display unexpected interrupts

/* check for unexpected interrupt blocks */

          if ((dtireg(8) AND 1) = 0)
               display no interrupt blocks available

          else

/* determine last word in interrupt queue */

          if (dtireg(6) = start_int_que)
               end_int_que = start_int_que+ num_blks*3-1
          else
               end_int_que = dtireg(6) - 1
          endif

          do i=start_int_que, end_int_que,3
               if ((dtimem(i) AND 0x000e) ! = 0)
                    display unexpected interrupt block found
               endif
          enddo
     endif
```

8.6.3 Remote Terminal Example 1

The first remote terminal example enables one subaddress in the receive direction, one subaddress in the transmit direction, and one mode code with a data word. A single buffer is used in each case. No interrupts are generated. Upon completion of the buslist, the message status word is checked for errors and received data is displayed.

REMOTE TERMINAL EXAMPLE 1

```
/* define constants */

     mem_size = 16k or 32k, depending on switch setting
     resp_blks = 0x0400                  /* start of subaddress response blocks */
     ill_sa = 0x0600                     /* illegal subaddress */
     ill_mc = 0x6e00                     /* illegal mode code */

/* transmit subaddress info */

     xmt_sa = 1                          /* arbitrary transmit subaddress */
     xmt_cntl = 0                        /* enable subaddress, no intrpts, 1 bfr */
     xmt_data = 0x1000                   /* pointer to data area */
     xmt_stat = 0x1a00                   /* pointer to message status area */

/* receive subaddress info */

     rcv_sa = 2                          /* arbitrary receive subaddress */
     rcv_cntl = 0                        /* enable subaddress, no intrpts, 1 bfr*/
     rcv_data = 0x2000                   /* pointer to data area */
     rcv_stat = 0x2a00                   /* pointer to message status area */

/* mode code info */

     mc = 17                             /* arbitrary mode code with data */
     mc_cntl = 0x0400                    /* illegal transmit mode code */
     mc_data = 0x3000                    /* pointer to data area */
     mc_stat = 0x3a00                    /* pointer to message status area */
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Remote Terminal Example-1 is continued from the previous page.

```
/* base on-board memory and registers */

     base on-board memory
     if (base fails) terminate

     base on-board registers
     if (base fails) terminate

/* reset the board by reading register 18 (allow time for
   operation to complete*/

     dummy = dtireg(18)
     dummy = dtireg(1)

/* enable 112X memory */
     dtireg (18) = 1

/* clear on-board memory (mem_size is memory size in words) */

     do i = 0,mem_size-1
          dtimem(i) = 0
     enddo

/* disable all subaddresses and mode codes */

     do i=1,30
          dtimem(resp_blks + (i-1)*4) = ill_sa
          dtimem(resp_blks + 128 + (i-1)*4) = ill_sa
     enddo

     do i=0,15
          dtimem(resp_blks + 256 + i*4) = ill_mc
     enddo

/* enable receive subaddress and set msg status word to FFFF */

     dtimem(resp_blks + (rcv_sa-1)*4) = rcv_cntl
     dtimem(resp_blks + (rcv_sa-1)*4 + 1) = rcv_stat
     dtimem(resp_blks + (rcv_sa-1)*4 + 2) = rcv_data
     dtimem(rcv_stat) = 0xFFFF


/* enable transmit subaddress and set msg status word to FFFF */

     dtimem(resp_blks + 128 + (xmt_sa-1)*4) = xmt_cntl
     dtimem(resp_blks + 128 + (xmt_sa-1)*4 + 1) = xmt_stat
     dtimem(resp_blks + 128 + (xmt_sa-1)*4 + 2) = xmt_data
     dtimem(xmt_stat) = 0xFFFF

/* enable mode code response and set msg status word to FFFF */
     dtimem (resp_blks + 256 + 4*(mc % 16)) = mc_cntl
     dtimem (resp_blks + 256 + 4*(mc % 16) + 1) = mc_stat
     dtimem (resp_blks + 256 + 4*(mc % 16) + 2) = mc_data
     dtimem (mc_stat) = 0xFFFF

/* load arbitrary data values to transmit */

     wrd_cnt = number of words to transmit

     do i = 1,wrd_cnt
          dtimem(xmt_data+i-1) = i
     enddo
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Remote Terminal Example-1 is continued from the previous page.

```
/* load registers */

     dtireg(7) = 0                        /* disable all interrupts */
     dtireg(9) = 0
     dtireg(2) = resp_blks                /* define start of subaddress resp blocks */
     dtireg(10) = 1                       /* define rt address, no options */
     dtireg(13) = 0                       /* reset rt timer */
     dtireg(0) = 0x0181                   /* enable bus a and b and start rt*/

/* allow time for buslist to execute */

/* examine message status words to verify traffic was generated
   (the message status word could be FFFF if an error occurred
   during a non broadcast message of 31 words when the RT timer
   was FF and the subsystem failed, but this situation is unlikely
   to occur) */

     stat = dtimem(xmt_stat)
     if (stat = 0xFFFF)
          display no data transmitted
     else if ((stat AND 0xa000) != 0)
          display message error or subsystem failed)
     endif

     stat = dtimem(rcv_stat)
     if (stat = 0xFFFF)
          display no data received
     else if ((stat AND 0xa000) != 0)
          display message error or subsystem fail message
     else
          wrd_cnt = (stat AND 0x1F00) / 0x100
     if (wrd_cnt = 0) wrd_cnt = 32
          do i =1,wrd_cnt
               display dtimem(rcv_data+i-1)
          enddo
     endif

     stat = dtimem(mc_stat)
     if (stat = 0xFFFF)
          display no data received
     else if ((stat AND 0xa000) !=0)
          display message error or subsystem fail message
     else
          display dtimem(mc_data)
     endif
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8.6.4 Remote Terminal Example 2

The second remote terminal example enables all subaddresses in both directions and all mode codes according to MIL-STD-1553B. A single-buffering scheme is used in each case. Interrupts are generated when the receive subaddress is accessed or when the Synchronize With Data Word Mode Code (Mode Code 17) is transacted. Upon completion of the buslist, all entries on the interrupt queue are processed by checking the message status words for errors and displaying received data. (Since one buffer per subaddress is used, the last message sent to the specified subaddress or mode code is displayed when an interrupt block involving the specified receive subaddress or mode code is encountered).

REMOTE TERMINAL EXAMPLE 2

```
/*define constants */

     mem_size = 16k or 32k
     resp_blks = 0x0400                  /* start of subaddress response blocks */
     start_int_que = 0                   /* start of interrupt queue */
     num_blks = 255                      /* number of interrupt blocks */

/* transmit subaddress info */

     xmt_cntl = 0x100                    /* enable subaddress, no intrpts, 1 buffer */
     xmt_data = 0x1000                   /* pointer to data area */
     xmt_stat = 0x1a00                   /* pointer to message status area */

/* receive subaddress info */

     rcv_cntl = 0x100                    /* enable subaddress, no intrpts, 1 buffer */
     rcv_data = 0x2000                   /* pointer to data area */
     rcv_stat = 0x2a00                   /* pointer to message status area */

/* mode code info (interrupts enabled) */

     mc_cntl(1) = 0xcb00                 /* mode code 0 and 16 */
     mc_cntl(2) = 0x8500
     mc_cntl(3) = 0xcb00
     mc_cntl(4) = 0x8b00
     mc_cntl(5) = 0x8500                 /* mode code 4 and 20 */
     mc_cntl(6) = 0X8e00
     mc_cntl(7) = 0x8e00
     mc_cntl(8) = 0x8e00
     mc_cntl(9) = 0x8e00                 /* mode code 8 and 24 */
     mc_cntl(10) = 0x6e00
     mc_cntl(11) = 0x6e00
     mc_cntl(12) = 0x6e00
     mc_cntl(13) = 0x6e00                /* mode code 12 and 28 */
     mc_cntl(14) = 0x6e00
     mc_cntl(15) = 0x6e00
     mc_cntl(16) = 0x6e00
     mc_data = 0x3000                    /* pointer to data area */
     mc_stat = 0x3a00                    /* pointer to message status area */

/* base on-board memory and registers */

     base on-board memory
     if (base fails) terminate
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Remote Terminal Example-2 is continued from the previous page.

```
/* reset the board by reading register 18 (allow time for
   operation to complete) */

     dummy = dtireg(18)
     dummy = dtireg(1)

/* enable 112X memory */
     dtireg (18) = 1

/* clear on-board memory (mem_size is memory size in words) */

     do i = 0,mem_size-1
          dtimem(i) = 0
     enddo

/* assuming that interrupt blocks are contiguous, create interrupt
   block links for interrupt queue (remember that link of first
   block is two words after first word of the block and set link
   of last block point to first block) */

     do i = 1,num_blks-1
          dtimem(start_int_que + i*3-1) = start_int_que + i*3
     enddo

     dtimem(start_int_que + num_blks*3-1) = start_int_que

/* enable receive subaddresses and set msg status word to FFFF */

     do i = 1,30
          dtimem(resp_blks + (i-1)*4) = rcv_cntl
          dtimem(resp_blks + (i-1)*4 + 1) = rcv_stat + (i-1)
          dtimem(resp_blks + (i-1)*4 + 2) = rcv_data + (i-1) *32
          dtimem(rcvf_stat + i - 1) = 0xFFFF
     enddo

/* enable transmit subaddresses and set msg status wrd to FFFF*/

     do i = 1,30
          dtimem(resp_blks + 128 + (i-1)*4) = xmt_cntl
          dtimem(resp_blks + 128 + (i-1)*4 + 1) = xmt_stat + (i-1)
          dtimem(resp_blks + 128 + (i-1)*4 + 2) = xmt_data + (i-1) *32
          dtimem(xmt_stat + i - 1) 0xFFFF
     enddo

/* enable mode code and set message status word to FFFF */

     do i=0,15
          dtimem(resp_blks + 256 + i*4) = mc_cntl(i+1)
          dtimem(resp_blks + 256 + i*4 + 1) = mc_stat + i
          dtimem(resp_blks + 256 + i*4 + 2) = mc_data + i
          dtimem(mc_stat + i) = 0xFFFF
     enddo

/* load arbitrary data values to transmit */
/* load registers */
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Remote Terminal Example-2 is continued from the previous page.

```
     dtireg(6) = start_int_que          /* start of interrupt queue */
     dtireg(7) = 0x121                  /* enable standard interrupts*/
     dtireg(9) = 0x32                   /* illegal sa, illegal brcast, and me */
     dtireg(19) = 0                     /* clear interrupt counter */
     dtireg(2) = resp_blks              /* define start of sa response blocks */
     dtireg(10) = 1                     /* define rt address, no options */
     dtireg(13) = 0                     /* reset rt timer */
     dtireg(0) = 0x0181                 /* enable bus a and b and start rt */

/* allow time for buslist to execute */
/* check for unexpected interrupts */

     if ((dtireg(8) AND 0x0120) !=0) display unexpected interrupt


/* for each interrupt block, display transmitted or received data
   (or error message if error occurred) - since single buffer
   scheme was used, the most recent received message is displayed
   for each interrupt block relating to a particular receive
   subaddress or receive mode code with data */

          if ((dtireg(8) AND 1) ! = 1)
               display no interrupt blocks available
          else

/* determine last word in interrupt queue */

               if dtireg(6) = start_int_que
                    end_int_que = start_int+que + num_blks *3-1
               else
                    end_int_que = dtireg(6) - 1
               endif

/* examine interrupt blocks one at a time */

          do i=start_int_que,end_int_que,3

/* only subaddress event and mode code event interrupts are
   expected */

               if ((dtimem(i) AND 0x7F3F) ! = 0)
                    display unexpected interrupt block
               else

/* get address of subaddress/mode code block */

                    blk_addr = dtimem(i+1)

/* get status word */

                    stat = dtimem(dtimem(blk_addr+1))
                    if (stat = 0xFFFF)
                         display no status response message
                    else if ((stat AND 0xa000) ! = 0)
                         display msg err or subsystem fail
                    else
/* get word count */

                         wrd_cnt = (stat AND 0x1f00) / 0x100
                    if (wrd_cnt = 0) wrd_cnt=32
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Remote Terminal Example-2 is continued from the previous page.

```
/* display data */

                    if (blk_addr < resp_blks + 128)
                    display received data
                    else if (blk_addr < resp_blk + 256)
                    display transmitted data
                    else
                    display mode code data
                    endif
               endif
          endif
     enddo

/* stop rt operation */

     dtireg(12) = 0
```

8.6.5 Remote Terminal Example 3

The third remote terminal example enables one subaddress in the receive direction. A specified number of buffers are reserved for messages. When the last message is received, an interrupt is generated. The remote terminal operation is halted and the received data is displayed (or an error message is displayed if an error occurred).

REMOTE TERMINAL EXAMPLE-3

```
/* define constants */

     mem_size = 16k or 32k
     resp_blks = 0x0400                  /* start of subaddress response blocks*/
     start_int_que = 0                   /* start of interrupt queue*/
     num_blks = 255                      /* number of interrupts blocks*/
     rcv_sa = 2                          /* arbitrary receive subaddress*/
     num_msg = 7                         /* index value*/
     rcv_cntl = 0x0080 OR num_msg        /* interrupt when index = 0 */
     rcv_data = 0x2000                   /* receive data area*/
     rcv_stat = 02a00                    /* receive status area*/
     ill_sa = 0x6000                     /* illegal subaddress*/
     ill_mc = 0x6e00                     /* illegal mode codes*/

/* base on-board memory and registers*/

     base on-board memory
     if (base fails) terminate

     base on-board registers
     if (base fails) terminate

/* reset the board by reading register 18 (allow time for operation to complete) */

     dummy = dtireg(18)
     dummy = dtireg(1)

/* enable 112X memory */
     dtireg (18) = 1

/* clear on-board memory (mem_size is memory size in words) */
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Remote Terminal Example-3 is continued from the previous page.

```
     do i = 0,mem_size-1
          dtimem(i) = 0
     enddo

/* assuming that interrupt blocks are contiguous, create interrupt
   block links for interrupt queue (remember that link of first
   block is two words after first word of that block and let link
   of last block point to first block) */

     do i = 1, num_pkts-1
          dtimem(start_int_que + i*3-1) = start_int_que + i*3
     enddo

     dtimem(start_int_que + num_blks*3 - 1) = start_int_que

/* disable all subaddresses and mode codes*/


     do i = 1,30
          dtimem(resp_blks + (i-1)*4) = ill_sa
          dtimem(resp_blks + 128 + (i-1)*4) = ill_sa
     enddo

     do i = 0,15
          dtimem (resp_blks + 256 + i*4) = ill_mc
     enddo

/* enable receive subaddress and set msg status words to FFFF*/

     dtimem(resp_blks + (rcv_sa-1)*4 = rcv_cntl
     dtimem(resp_blks + (rcv_sa-1)*4 + 1) = rcv_stat
     dtimem(resp_blks + (rcv_sa-1)*4 + 2) = rcv_data

     do i=,num_msg
          dtimem(rcv_stat + i -1) = 0xFFFF
     enddo

/* load registers*/

     dtireg (6) = start_int_que          /* start of interrupt queue*/
     dtireg (7) = 1                      /* enable standard interrupts*/
     dtireg (9) = 0                      /* interrupt only when index = 0*/
     dtireg (19) = 0                     /* clear interrupt counter*/
     dtireg (2) = resp_blks              /* define start of sa response blocks*/
     dtireg (13) = 0                     /* reset rt timer*/
     dtireg (10) = 1                     /* define rt address, no options*/
     dtireg (0) = 0x0181                 /* enable bus a and b and start rt*/

/* wait for specified number of messages to this subaddress to be
   transacted or for another task (not shown) to set a flag after
   a specified period of time*/

     do while ( ((dtireg(8) AND (NOT (stop_task)) )
     enddo

/* halt rt operation */

     dtireg(12) = 0

/* indicate if all messages not received in specified time*/

     if (stop_task)
          display time-out message
     endif
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Remote Terminal Example-3 is continued from the previous page.

```
/* display any captured messages*/

     index = dtimem(resp_blks + (rcv_sa-1)*4) AND 0x7F
     if (index = num_msg)
          display no messages received
     else

/* point to first data word of first valid message*/

          current_data = rcv_data

/* examine messages actually received (index becomes 0 when
   seventh message received, so current data pointer and message
   status pointer point to message received after these first
   seven)*/

          current_data = rcv_data
          do i=1,num_msg-index
               if (dtimem(rcv_stat+i-1) = 0xFFFF
                    display no status received
               else if ((dtimem(rcv_stat+i-1) AND 0xa000) ! = 0
                    display msg error or subsystem fail
               else

/* get word count from message status word*/

                    wrd_cnt = (dtimem(rcv_stat+i-1)AND0x1F00)/0x100
                    if (wrd_cnt= 0) wrd_cnt = 32

/* display each data word of message */

                    do j=1,wrd_cnt
                         display dtimem(current_data + j -1)
                    enddo
                    endif

/* point to next word in receive area*/


               current_data = current_data + wrd_cnt

          enddo
     endif
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

8.6.6 Bus Monitor Example 1

The first bus monitor example lists the steps necessary to log all bus traffic. No interrupts are set. All monitor command blocks are stored in one area of memory and all MIL-STD-1553B data words are stored in another. After all traffic is recorded, the messages are output. (This example assumes that no more than "num_blks" messages will be transacted and that no message block overwriting will occur.)

BUS MONITOR EXAMPLE-1

```
/* define constants*/

     memory_size = 16K or 32K, depending on DTI-1120 switch setting
     start_msg = 0x0300                 /* message record storage begins here*/
     num_blks = 100                     /* there is room for more messages*/
     start_data = 0x1000                /* data word storage begins here*/
     num_wrds = 32                      /* max number of data words per message*/

/* base on-board memory and registers*/

     base on-board memory

     if (base fails) terminate

     base on-board registers

     if (base fails) terminate

/* reset the board by reading register 18 (allow time for
   operation to complete) */

     dummy = dtireg(18)
     dummy = dtireg(1)

/* enable 1120 memory */
     dtireg(18) = 1

/* clear on-board memory (memory_size is memory size in words)*/

     do i=0, memory_size - 1
          dtimem (i) = 0
     enddo

/* initialize the data pointer and the tail pointer in the message
   record block (the control word was set to zero when memory was
   cleared)*/

     do i =1, num_blks
          addr = start_msg + (i-1) * 7
          dtimem(addr+3) = start_data + (i-1) * num_wrds /* data pointer*/
          dtimem(addr+6) = addr + 7     /* tail pointer*/
     enddo

/* load registers*/

     dtireg (7) = 0                     /* disable interrupts*/
     dtireg (9) = 0
     dtireg (2) = start_msg             /* point to first msg record blk*/
     dtireg(14) = 0xc000                /* log all and bus monitor select*/
     dtireg(0) = 0x181                  /* enable both buses and start*/

/* wait for end of traffic*/
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Monitor Example-1 is continued from the previous page.

```
/* output data*/

     done = FALSE
     addr = start_msg                          /* point to first message*/
     do while (NOT done)

          cntl = dtimem(addr)                  /* get control word of this block*/
          if ((cntl AND 0x8000)                /* see if data stored in this block*/
               done = TRUE
          else
          if ((cntl AND 0x4000) == 0x4000)     /* see if message error bit is set*/
               display message error occurred
          if ((cntl AND 0x1000) ==0x1000)      /* determine bus used in transfer*/
               bus = A
          else
               bus = B
          endif
          tag = cntl AND 0xff                  /* get time tag info*/
          display bus, tag

          cmd1 = dtimem(addr+1)                /* get command word 1 of this block*/
          rt = (cmd1 AND 0xf800)/0x800         /* get rt*/
          dir = cmd1 AND 0x400                 /* get direction*/
          if (dir == 0)
               drctn = RCV
          else
               drctn = XMT
          endif
          sa = (cmd1 AND 0x3e0)/0x20           /* get subaddress*/
          display rt, drctn, sa

          wc_or_mc = cmd1 AND 0x1f             /* get word count/mode code*/

          if (sa == 0) OR (sa == 31)           /* determine if mode subaddress*/
               if (wc_or_mc<16)
                    wrd_cnt=0                  /* mode code without data*/
               else
                    wrd_cnt=1                  /* mode code with data*/
               endif
          else
               wrd_cnt=wc_or_mc                /* get word count of data transfer*/
          endif

          cmd2 = dtimem(addr + 2)
          if (cmd2 ! = 0)                      /* is there a second command?*/
               rt = (cmd2 AND0xf800)/0x800     /* get second rt*/
               drctn = XMT                     /* by definition of rt-rt command*/
               sa = (cmd2 AND 0x3e0)/0x20      /* get sa of second command*/
               display rt, drctn, sa
          endif
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Monitor Example-1 is continued from the previous page.

```
          if (wrd_cnt! = 0)
               data_ptr = dtimem(addr + 3)   /* get data pointer*/
                    do i=1,wrd_cnt
                    display dtimem(data_ptr+i-1)  /* retrieve and display data words*/
                    enddo
          endif

          display dtimem(addr+4)        /* display status word*/

          if (dtimem(addr+5)! = 0)      /* if rt-rt, display second status*/
               display dtimem(addr+5)
          endif
               addr = dtimem(addr+6)    /* get addr of next msg record blk*/
     enddo
```

8.6.7 Bus Monitor Example 2

The second bus monitor example lists the steps necessary to log bus traffic involving selected remote terminals. An interrupt is generated when every 200th message record block is loaded with a message. The most recent 200 monitor command blocks can be written to a disk. The data words associated with a message are stored immediately after the monitor record block containing the command and status words for this message.

BUS MONITOR EXAMPLE-2

```
/* define constants*/

     memory_size = 16k or 32k, depending on DTI-1120 switch setting
     start_msg = 0x0300                 /* message record storage begins here*/
     num_blks = 400                     /* 16k board; can store 800 messages on
                                           32k board*/

     num_wrds = 32                      /* max number of data words per message*/
     blk_size = 200                     /* interrupt every "blk_size" captured
                                           messages*/
     num_blks = 255                     /* number of interrupt blocks*/

/* base on-board memory and registers*/

     base on-board memory

     if (base fails) terminate

     base on-board registers

     if (base fails) terminate

/* reset the board by reading register 18 (allow time for
   operation to complete) */
     dummy = dtireg(18)
     dummy = dtireg(1)

/* Enable 1120 memory*/
     dtireg(18) = 1
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Monitor Example-2 is continued from the previous page.

```
/* clear on-board memory (memory_size is memory size in words)*/
     do i=0, memory_size-1
          dtimem(i) = 0
     enddo

/* assuming that interrupt blocks are contiguous, create interrupt
   block links for interrupt queue (remember that link of first
   block is two words after first word of that block) and let link
   of last block point to first block*/

     do i=1, num_blks-1
          dtimem(start_int_que + i*3-1) =start_int_que + i*3
     enddo
     dtimem(start_int_que + num_blks*3-1) = start_int_que

/* initialize the data pointer and the tail pointer in the monitor
   control block (the control word was set to zero when memory was
   cleared) */

     do i=1, num_blks
          addr = start_msg + (i-1)*39
          if (MOD(addr, 200) == 0) dtimem(addr) = 0x2000 /*int bit in cntrl wrd*/
          dtimem(addr+3) = addr + 7          /* data pointer*/
          dtimem(addr+6) = addr + 39         /* tail pointer*/
     enddo

/* open disk file where data will be written*/
     open disk file

/* load registers*/

     dtireg(6) = start_int_que               /* define start of interrupt queue*/
     dtireg(7) = 1                           /* enable standard priority intrpts*/
     dtireg(9) = 1                           /* interrupt when data placed in*/
                                             /* msg rec blk with int bit set*/
     dtireg(19) = 0                          /* clear interrupt block counter*/
     dtireg(2) = start_msg                   /* point to first msg record blk*/
     dtireg (14) = 0xa000                    /* log all and bus monitor select*/
     dtireg(16) = 0x0401                     /* monitor RT0, RT10,*/
     dtireg(17) = 0x4010                     /* RT20 and RT30*/
     dtireg(0) = 0x181                       /* enable both buses and start*/

/* save 200 messages to disk when an interrupt is set (the only
   interrupt that is set is the "block accessed" interrupt) -
   assume a Control-C handler exists that sets a flag when a
   Control-C is entered at the keyboard*/
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

This Bus Monitor Example-2 is continued from the previous page.

```
cntl_c_flag = FALSE
addr = start_msg
do while (NOT cntl_c_flag)

     if (dtireg(8) ! = 0)                    /* has interrupt occurred?*/
          dtireg(8) = 1                      /* clear interrupt*/
          write dtimem(addr)through dtimem(addr+blk_size-1) to disk
          dtimem(addr) = 0                   /* clear message record block*/
          dtimem(addr+1) = 0                 /* (except pointers) so that*/
          dtimem(addr+2) = 0                 /* new message can be detected*/
          dtimem(addr+4) = 0                 /* when it is stored*/
          dtimem(addr+5) = 0
          addr = addr + blk_size             /* point to start of next block*/
     if (addr>num_blks)addr=start_msg        /* after writing last block,*/
                                             /* point to first block*/
     endif
enddo

if (current buffer contains data) write current buffer to disk

close disk file
```

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

9.0 DIAGNOSTICS

9.1 Overview

The 112X Diagnostics software provides testing to determine that the 112X boards are working correctly with the PC-AT host. This Diagnostics software requires DOS 3.0 or higher on a PC-AT computer or compatible system. To run correctly, at least one but no more than eight 112X boards must be installed in the host computer.

The 112X PC-AT Diagnostics program reads and writes to memory and registers and tests the system's interrupt mechanism, memory and registers. The program also verifies correct operation of on-board MIL-STD-1553B bus transceivers via a built-in self test. Diagnostics software performs hardware tests controlled by user-selectable commands. These commands, along with tests and access facilities, are described in this chapter.

The four test categories that verify the 112X hardware is operating properly are listed below:

- Memory Test
- Register Test
- Interrupt Test
- Built-In Test

The Memory Test has seven user-defined, selectable self-tests, that verify proper on-board memory (RAM) operation. When the Diagnostics program is invoked, all memory tests are enabled by default.

The Register Test consists of two subtests that are not user-selectable.

The variety of commands allows manipulation of these tests to suit the user's needs. Any combination of the tests may be conducted using the **TEST** command. When specific tests are enabled, the user executes them by issuing the **RUN** command. The enabled tests execute until one of the following events occurs:

- A user-specified pass count (REPEAT) is reached
- A user-specified error count (MAXERR) is exceeded
- The user enters a 'Ctrl-C' at the keyboard

9.2 Tests

9.2.1 Test 1 — RAM Address Test

Test 1 loads each memory location with a value equal to its memory address. The values are then read and compared to their respective memory addresses. The error count is incremented by one for each mismatch that is found.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

9.2.2 Test 2 — Walking 0's Test

Test 2 checks each memory location's ability to have a single bit OFF. A data word is written to each memory location containing only one '0' bit, then the word is read and verified. The '0' bit is first positioned as the least significant bit and then "walked" through each of the 16-bit positions. For any inconsistency encountered, the error count is incremented accordingly.

9.2.3 Test 3 — Walking 1's Test

Test 3 checks each memory location's ability to have a single bit ON. Test 3 operates in the same manner as Test 2. The process is accomplished by writing data words to memory locations containing only one '1' bit.

9.2.4 Test 4 — All Bits Test

Test 4 loads each RAM location with the value 'FFFF' *hex*, and reads each location to verify proper operation. If any locations contain a bit OFF, the error count is incremented.

9.2.5 Test 5 — Value AAAA Test

Test 5 loads each 112X RAM location with the value 'AAAA' *hex* and reads each location to verify proper operation. If any locations contain a value other then AAAA, the error count is incremented.

9.2.6 Test 6 — Value 5555 Test

Test 6 loads each 112X RAM location with the value '5555' *hex* and reads each location to verify proper operation. If any locations a contain a value other than 5555, the error count is incremented.

9.2.7 Test 7 — Value 0000 Test

Test 7 loads each 112X RAM location with the value '0000' *hex* and reads each location to verify proper operation. If any locations contain a bit ON, the error count is incremented.

9.2.8 Test 8 — Register Test

Test 8 verifies the PC's ability to correctly address, write to and read from the 112X's registers.

READ/WRITE INTERNAL REGISTERS:

The 112X contains 14 internal registers in common. The Diagnostics program writes a value of 'AAAA' *hex* to three read/write registers common to both products and reads the values to verify functionality. The error count is incremented for each mismatch that is found.

READ/WRITE OTHER REGISTERS:

The 112X houses three additional registers.

Two of these registers read and clear the interrupt packet count of standard level interrupts. These registers are tested by generating a defined number of interrupts and verifying the interrupt count. Several interrupts are then de-queued to verify that the

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

decrement interrupt count register works properly. Proper operation of the interrupt count register exceed bit is verified by generating more than 255 interrupts.

The third additional register can perform a full hardware reset, enable and disable interrupts, enable and disable MIL-STD-1553B transceiver operation, and execute a built-in self test. This register is tested by loading certain registers with select values and placing the 112X in a run state, without generating any MIL-STD-1553B traffic.

All three of these additional registers are verified for reset after the master reset register is accessed.

9.2.9 Test 9 — Interrupt Test

Test 9 verifies that the hardware interrupt mechanism operates properly by resetting the hardware, clearing memory and enabling a BC Instruction Execution Interrupt and an End Of List Interrupt. The bus instruction is then executed to verify that the interrupts have been generated and recognized by the host, causing a valid interrupt packet to be placed in the interrupt queue.

9.2.10 Test 10 — Transceiver Built-In Test

Test 10 verifies correct operation of the on-board MIL-STD-1553B bus transceivers. When invoked, the 112X hardware is configured in the Remote Terminal operation mode. A hard-coded command word is issued onto the bus from the hardware with transceivers disabled. This command is received by the RT and verified. The same command is reissued with transceivers enabled. This test is repeated using a different command word and the results are verified. This test also verifies the hardware's ability to receive valid bus commands.

The results of the Test 10 are displayed to the terminal.

9.3 Using The Diagnostics Software

To invoke the Diagnostics program, type **DIAG** at the DOS prompt. After displaying the program name and version number, the following prompt is displayed:

```
Diagnostics>
```

The following notation is used to describe commands:

CAPITAL LETTERS: enter exactly as shown

small letters: enter a value

[items in brackets]: entry of item is optional

All values should be entered in decimal form at the command line. Commands must be terminated by pressing the <return> key. Either upper or lower case may be used for alphabetic characters. If the user wishes to enter a value in hexadecimal form, a double quote (") must precede the data value.

Artisan Technology Group - Quality Instrumentation ... Guaranteed | (888) 88-SOURCE | www.artisantg.com

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP