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 *Abaco Systems / SBS ABI-PC2* at our website: ***Click HERE***

1553 ADVANCED BUS INTERFACE FOR PC (ABI-PC2)

REFERENCE MANUAL

Microcode Load for ABI Function
Microcode Product Number: 0118
Microcode Version:

Microcode Loads for PASS-1000 Function
Microcode Product Number: 0903
Microcode Version:

Microcode Product Number: 0904
Microcode Version:

August 15, 1994

Document Number ABI-PC2-94227

ERRATA

This section details corrections to the following documents:

ABI-PC2 Reference Manual, January 18, 1995 (see cover page for current date)
ABI-V3 Reference Manual, August 15, 1994
ABI-Q Reference Manual, August 15, 1994
ABI-V2 Reference Manual, August 15, 1994
ABI-PC/AT Reference Manual, August 15, 1994
ABI-V4 Reference Manual, December 15, 1995

ALL DOCUMENTS

Built-In Test Routines

The procedure for BIT #3 should read as follows:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)
2. Set MAP register by loading offset 0010h with "3". (Set MAP)
3. Load offset 003Fh with any value.
4. Load offset 0040h with "0000".
5. Start ABI by loading offset 0000h with "0240" followed by loading offset 0000h with "0242". NOTE: If running this test on a PASS-1000, load offset 0000h with "0040", followed by "0042".
6. Load starting test address in offset 0044h. Minimum acceptable value is "0080".
7. Load ending test address in offset 0045h. Maximum acceptable value is "FFFF".
8. Start the test by writing "0001" to the Command (CMD) word at offset 0040h.
9. Wait for offset 0040h to equal "0000".
10. Check offset 0043h. If a nonzero value is present at this offset, errors occurred during the test.

ABI Operations Overview

Error Table

Add error code 15E to this table. This error occurs when an RT sends an extra data word.

ABI-PC2 & ABI-V3 ONLY

The information contained in the following paragraphs applies only to products having the following microcode loads:

ABI-V3	M019H and above
ABI-PC2	M025E and above

The microcode load is listed on the product box label and is also printed on the labels attached to the actual PROMs on the board. If you are unable to determine the microcode load for your SBS product, contact SBS Technologies at 1-800-SBS-1553 or (505) 875-0600.

Built-In Test Routines

Microcode Organization, second paragraph, page 3-1

For products having the microcode loads listed above, the Built-In-Test (BIT) microcode is found only in the high PROM area. Select a specific BIT routine by calling one of seven starting addresses, listed in Table 3-1.

Table 3-1 BIT Addresses

Microprogram Address	Program Selected	Description
0001h	Built-In-Test # 1	Tests Internal registers and addressing modes
0002h	Built-In-Test #2	Tests ALU and condition code operation
0003h	Built-In-Test #3	Tests RAM with Read/Write/Verify pattern-specific data
0004h	Built-In-Test #4	More testing of internal registers and addressing modes
0005h	Built-In-Test #5	Tests on-board timer sub-system
0006h	Built-In-Test #6	Tests 1553 interface logic and front end
0007h	Built-In-Test #7	Tests external discrete trigger function

Built-In Tests 1-7, pages 3-3 through 3-12

Each BIT test includes a "Start" instruction which involves loading offset 0000h with either "0242" or "0042." For products having the microcode loads listed above, load offset 0000h with "0242."

Full Function Programmer's Reference

Microcode Initialization, page 4-19

When an ABI module is powered-up or reset, the host software must perform the proper startup procedure to initialize the microcode program.

Table 4-6 Microcode Program Identification

Microprogram ID #		Microprogram Description	Where Discussed
1	0001h	Reserved	-------
2	0002h	Reserved	
3	0003h	Reserved	
4	0004h	Reserved	
5	0005h	Reserved	
6	0006h	Reserved	
7	0007h	Reserved	
8	0008h	Reserved	
9	0009h	Reserved	
10	000Ah	Reserved	
11	000Bh	Reserved	
12	000Ch	Reserved	
13	**000Dh**	**Full Function 1553B**	**"1553 Microcode Program Startup Procedure" in this manual subsection**
14	000Eh	Reserved	-------
15	000Fh	Reserved	

The Application Main Body program is the entry point for the "Full Function 1553B" microcode program that executes the BC, RT, Monitor, and Interrupt functions of this interface.

For products having the microcode loads listed above, the Built-In-Test routines are contained in the upper half of the PROM and application microcode in the lower half. To run the application microcode, select the lower PROM area (set CSR bit 9 to "0") in the "1553 Microcode Program Startup Procedure" which follows in this subsection.

INTRODUCTION

The Advanced Bus Interface (ABI) is a single card, real-time MIL-STD-1553 interface module for PC and VME host systems. The ABI provides the host system an advanced 1553 interface, including the following features: simultaneous and independent simulation of Bus Controller and 31 Remote Terminals, real-time monitoring, double-buffered monitoring and flexible event Interrupt capabilities. The ABI may be configured for one or more of these functions.

This manual provides detailed information on ABI setup and operations and is organized according to the following table (continued on following page).

Table 1-1 ABI Manual Organization

Section	Description
Introduction	Provides a description of the Reference Manual organization, and a discussion of the relationship between the Reference Manual and Interface Libraries Manual.
Hardware Installation	This section describes the steps required to install the ABI into the host system. For VME systems, the proper system address, interrupt level and interrupt vector must be set on the ABI board. The user may also set hardware jumpers to configure ABI system clock sources (internal ABI clock or external clock source). Switch and jumper settings are explained in this section. There is also a general discussion on configuring the ABI for short stub (1553A typically) or long stub (1553B) configurations, and the proper physical connections of the ABI to a 1553 bus.
Built-In-Test Routines	This section describes the various ABI Built-In-Test (BIT) routines that provide a comprehensive diagnostic utility for the ABI. Each test description explains the functionality, execution procedure and expected results for the BIT.

Table 1-1 ABI Manual Organization	
Section	**Description**
Full Function Programmer's Reference	This section provides a detailed overview of the ABI architecture and operations. The section includes an introduction and five subsections, as described below: **Introduction/ABI Operations Overview** - Provides a general system description and details the memory organization of the ABI. **Microcode Init** - Explains the procedure the host computer must follow to properly initialize ABI operations. **BC Simulation** - Describes the ABI Bus Controller functions and the related data structures. **RT Simulation** - Describes the ABI Remote Terminal functions and the related data structures. **Bus Monitoring** - Describes the two types of 1553 bus monitoring available with the ABI (Sequential and Map Monitoring) and the related data structures. The data structures used for Map Monitoring overlap those used for RT Simulation; therefore, both subsections must be read to fully understand Map Monitoring. **Interrupts** - Describes the action taken by the host to cause an ABI interrupt. This subsection also details the proper procedure for servicing both hardware and software polling interrupts.

Most of the procedures described in the *BIT Routines* section and *Full Function Programmer's Reference* section for setting-up and running ABI 1553 functions are provided as C functions or Ada procedures in the Interface Libraries. For example, the "1553 Microcode Program Startup Procedure" described in the *Full Function Programmer's Reference* can be performed by host software through one of two C function calls: "init_device()" or "start_microcode()". There are many defined library functions provided to simplify the host integration effort.

ABI users should first read the *Full Function Programmer's Reference* section, followed by the Interface Libraries Manual.

ABI-PC2 Hardware Installation
TABLE OF CONTENTS

TABLE OF FIGURES AND TABLES

ABI-PC2 HARDWARE INSTALLATION

Introduction

The ABI-PC2 is a single-board MIL-STD-1553 interface for an IBM PC/AT, or IBM compatible computer system. This section describes the unpacking, configuration, and installation of an ABI-PC2 into a host computer. Because proper installation is critical to the successful operation of the ABI-PC2, SBS Engineering recommends that the steps described here be followed explicitly. The steps involved in the installation include:

- Unpacking and Inspecting the Equipment
- Setting the ABI-PC2 RAM Base Address in PC Memory Space
- Setting the ABI-PC2 I/O Base Address in PC I/O Space
- Setting the ABI-PC2 Interrupt Level
- Selecting Long or Short Stub Connection
- Installing the Configured Board into the PC System
- Testing the Installed Board

> Note: This is an electronic product that is sensitive to electrostatic discharge. Normal precautions should be observed in handling the board to prevent damage.

Operational and Storage Specifications

- Maximum Power Consumption:
  - 5 volts at 2.5 amps
  - 12 volts at 400 milliamps
- Operating Temperature: 0° to 70° Celsius, at less than or equal to 95% humidity
- Storage Temperature: -65° to 150° Celsius, at less than or equal to 95% humidity
- Shock requirements are to the highest commercial standards
- Mean Time Between Failures: 72,000 hours, based on MIL-HDBK-217F

Front Panel Connectors

Front Panel Connectors			**Mating Connectors**		
Location	Part Number	Manufacturer	Pigtails	Part Number	Manufacturer
P4	CA2047	Milestek	Bus A & B	PL75	Trompeter
			trigger, IRIG	mating connector dependent on cable	

SBS sells bus configurations, part numbers BUS-2 (2-stub coupler) and BUS-3 (3-stub coupler), for transformer coupled systems. Contact SBS at 505-345-5353 for more information.

Other components are available from:

MilesTek
1 Lake Trail Drive
Argyle, Texas 76226
Attn: Frank or Jeanette Miles
800-524-7444
800-683-7444

MilesTek also provides the following:

twin-axial cable	CA2009-xxx (xxx denotes length)
1553 bus terminator	10-06403-025

Unpacking and Inspecting the Equipment

The ABI is shipped in a single box that contains the ABI board, cable adapter, and this manual. Carefully unpack the board and inspect it for any visible damage that could have occurred during shipment. If there is visible damage, contact SBS Engineering. When calling SBS, have the board and board serial number available.

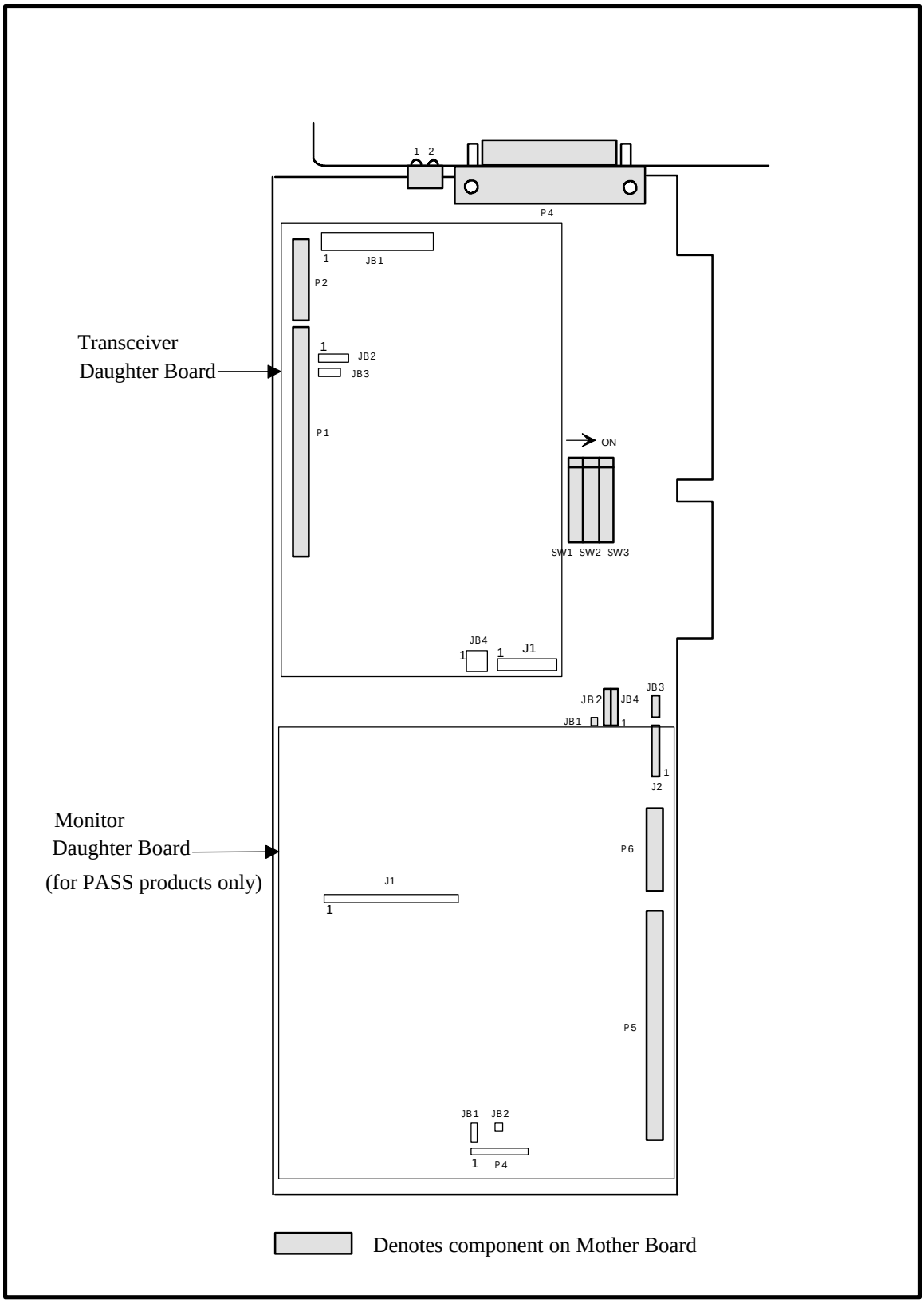


Figure 2-1 ABI-PC2 Board Layout

Setting the ABI-PC2 RAM Base Address in PC Memory Space

The ABI-PC2 contains 128 kilobytes (64 kilowords) of on-board RAM. This RAM may either be configured as one 128 kilobyte block or as two overlayed 64 kilobyte blocks at the same address. The base address of the board can be assigned any address that is on a 128 kilobyte boundary. Before setting the base address, study the unused memory blocks in the host system to determine if the board should be configured with a block of length 128 or 64 kilobytes.

The memory map of a PC-AT consists of the DOS memory area and the Extended memory area. If the ABI-PC2 is mapped into the DOS area, the board can be accessed using readily available software compilers. The drawback to mapping the board into the DOS area is that the amount of unused memory space is limited. The drawback to using the extended area is that the board can only be accessed using extended memory device drivers or compilers that run in 32-bit protected mode. If the ABI is mapped into this area and the user wishes to use a standard compiler, a device driver must be used to access the board. The use of a device driver causes each access of the board to be considerably slower than if the board is mapped into the DOS area or if a 32-bit protected mode compiler is used.

The DOS memory area consists of the addresses from 000000h to 100000h. Within this area, certain addresses are reserved by DOS and must not be used by the ABI-PC2. Use the DOS DEBUG program to read the memory block and determine if it is unused. (If all of the data in the block is FFh, the block is probably unused.) Typical good DOS addresses for the ABI-PC2 are 0D0000h and 0E0000h.

The Extended memory area consists of the addresses from 100000h to FFFFFFh. The ABI-PC2 may be installed at any address within this range, as long as it is **above the RAM configuration of the machine**. Therefore, if the extended RAM area of the PC system is full, typically 32 megabytes, the upper 16 megabytes must be removed prior to mapping the ABI-PC2 in extended memory.

The decision of where to map the ABI is up to the user and should be determined based on the memory spaces available, the type of compiler to be used, and the speed needed for the application. The factory default base address for the ABI-PC2 is F00000h, in extended memory.

Table 2-1 contains the memory map of the PC/AT, and should be used to help determine a possible base address for the ABI-PC2.

Table 2-1 PC/AT Memory Map		
Address Range	Length	Description
000000h - 0BFFFFh	768K Bytes	Reserved for DOS.
0C0000h - 0CFFFFh	64K Bytes	Unused. May be used by video adaptors.
0D0000h - 0DFFFFh	64K Bytes	Unused. This block may be used by certain adaptors or by an Expanded Memory Manager (EMM). This block can only be a base address if the board is configured in the 64K byte block mode.
0E0000h - 0EFFFFh	64K Bytes	DOS ROM area. May be unused in an IBM compatible computer.
0F0000h - 0FFFFFh	64K Bytes	DOS ROM area.
100000h - FFFFFFh	15M Bytes	Extended Memory area. Some or all of this area may be used by extended memory installed in the system.

Setting the ABI-PC2 Base Address

Switch SW3 is used to select the base address of the ABI-PC2. If the desired value of the address line is "0", the switch should be placed in the ON position; if the desired value is "1", the switch should be placed in the OFF position. Figure 2-2 illustrates the switch positions for the factory default base address (F00000h). The switch is read from left to right with SW3-8 being the least significant position. Table 2-2 shows the relationship between the switch positions and the address lines as well as an example of the default base address setting. Note: Bits 0 through 15 are not selectable and default to zeros.

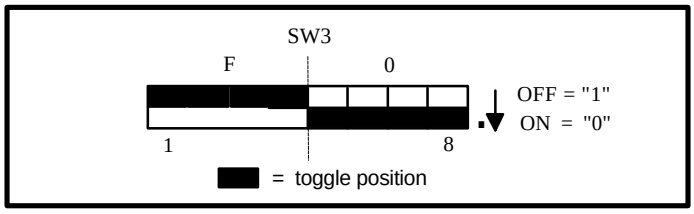


Figure 2-2 Default Base Address Switch Positions

Table 2-2 Default Base Address Selection			
Address Bit	Switch	Desired Value	Switch Setting
A23	SW3-1	1	OFF
A22	SW3-2	1	OFF
A21	SW3-3	1	OFF
A20	SW3-4	1	OFF
A19	SW3-5	0	ON
A18	SW3-6	0	ON
A17	SW3-7	0	ON
A16	SW3-8	0	ON

Figure 2-3 and Table 2-3 show the switch positions for the DOS memory base address 0D0000h.

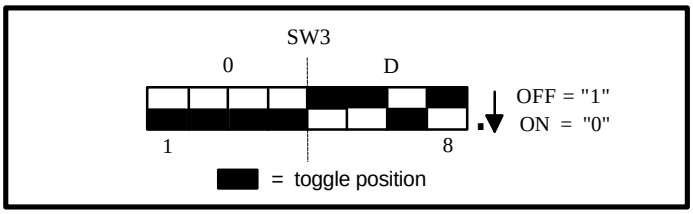


Figure 2-3 Switch Positions for Base Address in DOS Space

Table 2-3 Base Address in DOS Space

Address Bit	Switch	Desired Value	Switch Setting
A23	SW3-1	0	ON
A22	SW3-2	0	ON
A21	SW3-3	0	ON
A20	SW3-4	0	ON
A19	SW3-5	1	OFF
A18	SW3-6	1	OFF
A17	SW3-7	0	ON
A16	SW3-8	1	OFF

Setting the ABI-PC2 I/O Base Address in PC I/O Space

Switch SW1 is used to set the base I/O address of the ABI Control Register. The Control Register is used to control the following:

- Enabling the ABI-PC2
- Selecting between 128K Byte and 64K Byte Modes
- Selecting between upper or lower blocks (64K Byte mode only)

The I/O base address of the Control Register can be set to any value between 300h and 3FFh. The eight switches of SW1 correspond to the eight least significant address lines. Table 2-4 shows the correspondence between the switches and the address lines, along with an example for the default setting of 390h.

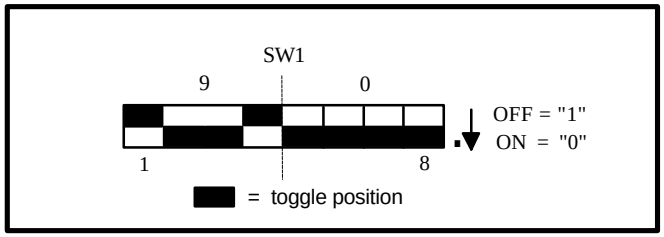


Figure 2-4 Default I/O Base Address Switch Positions

Table 2-4 I/O Base Address Selection

Address Bit	Switch	Desired Value	Switch Setting
A7	SW1-1	1	OFF
A6	SW1-2	0	ON
A5	SW1-3	0	ON
A4	SW1-4	1	OFF
A3	SW1-5	0	ON
A2	SW1-6	0	ON
A1	SW1-7	0	ON
A0	SW1-8	0	ON

Monitor Daughter Board

When the Monitor Daughter Board is installed (for use with PASS products), it takes up extra I/O address space. If installed in extended memory, it uses 128 kilobytes of address space immediately following the base address and I/O base address.

Setting the ABI-PC2 Interrupt Level

Switch SW2 is used to set the interrupt level used by the ABI-PC2. The user can choose between eight different interrupt levels or no interrupts. Table 2-5 contains a list of the possible interrupt level settings along with their use in an IBM PC/AT system. Only one of the switch positions in SW2 can be ON; if more are ON, undetermined results will occur. Also, if an interrupt level is selected that is used by another device, undetermined results will occur.

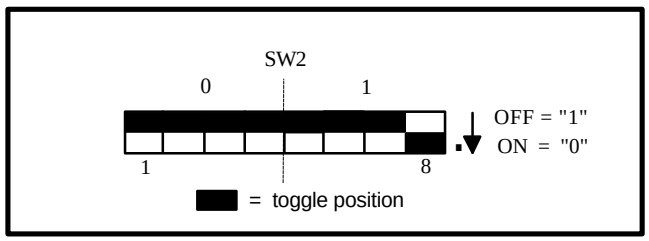


Figure 2-5 Example of ABI-PC2 Interrupt Level 15

Table 2-5 Interrupt Level Selection		
Switch (SW2)	Interrupt Level	Use in PC/AT
ALL OFF	NONE	
SW2-1	IRQ3	COM2. Second serial port.
SW2-2	IRQ5	LPT2. Second parallel port.
SW2-3	IRQ7	LPT1. First parallel port.
SW2-4	IRQ9	Unused.
SW2-5	IRQ10	Unused.
SW2-6	IRQ11	Unused.
SW2-7	IRQ12	Unused.
SW2-8	IRQ15	Unused.

External Connector Pinouts

Table 2-6 shows the pinouts for the external ABI-PC2 connector labeled P4.

Table 2-6 Connector P4 Pinout			
Pin	Signal	Pin	Signal
1	Primary Bus +	14	Primary Bus -
2	Secondary Bus +	15	Secondary Bus -
3	Reserved	16	Reserved
4	Reserved	17	Reserved
5	Reserved	18	Reserved
6	Reserved	19	Reserved
7	External Trigger *	20	GND
8	+5VDC	21	GND
9	External Clock	22	GND
10	External Enable *	23	GND
11	External Clear *	24	GND
12	Reserved	25	GND
13	+5VDC		
* active low signal			

Installing the ABI-PC2 into a PC/AT System

The next step is to install the ABI into the desired slot in the host system. To do this, follow the instructions supplied by the system manufacturer.

PASS-1000 Installation

If installing the ABI-PC2 for use with the PASS-1000, please refer to the "Installation" section of the PASS-1000 Reference Manual for software installation instructions.

Connecting the ABI to a 1553 Bus

For operation, the ABI must be connected to a MIL-STD-1553 bus. This is not necessary for initial testing of the board. For test purposes, it is sufficient to simply connect standard bus terminators to the A and B bus connectors on the front panel of the interface.

These MIL-STD-1553 bus connections must be made very carefully. **The ABI will not operate properly if the bus connections are not correct.**

The MIL-STD-1553 bus consists of a twisted-pair cable with terminators at each end, ranging in length from a few inches to several hundred feet. The physical connection of a subsystem to this bus is called a stub. In MIL-STD-1553 there are two types of stubs defined: 1) long stubs or 2) short stubs. Short-stub connections to a bus are rarely used. Long-stub connections are the most common and require several components. These components include a transformer and two resistors, used at the exact point where the connection is made to the bus. These components are typically enclosed in a small module called a *bus coupler*. These couplers may provide for connection of only one stub or several stubs. The coupler has two connectors for the bus (usually at opposite ends of the coupler) and a connector for each stub (typically mounted perpendicular to the bus connectors).

Subsystem stubs will not function properly if connected directly together. Proper coupling devices must be used for each stub and the bus must be properly terminated. Figures 2-6a and b illustrate the proper and improper connection schemes for long-stub coupling and the terminology used for the various components associated with the bus. A three-stub coupler is shown in the figure. In this case, the bus is contained entirely within the coupler. If other devices are to be connected to the bus, one of the terminators must be removed and replaced with a cable connecting this coupler to another coupler. The other coupler could then be terminated or the bus extended further.

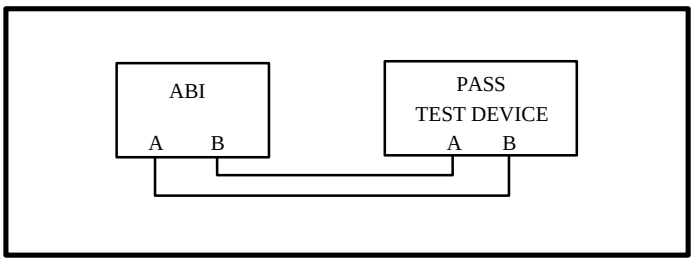


Figure 2-6a Incorrect Method of Bus Coupling (Long Stubs)

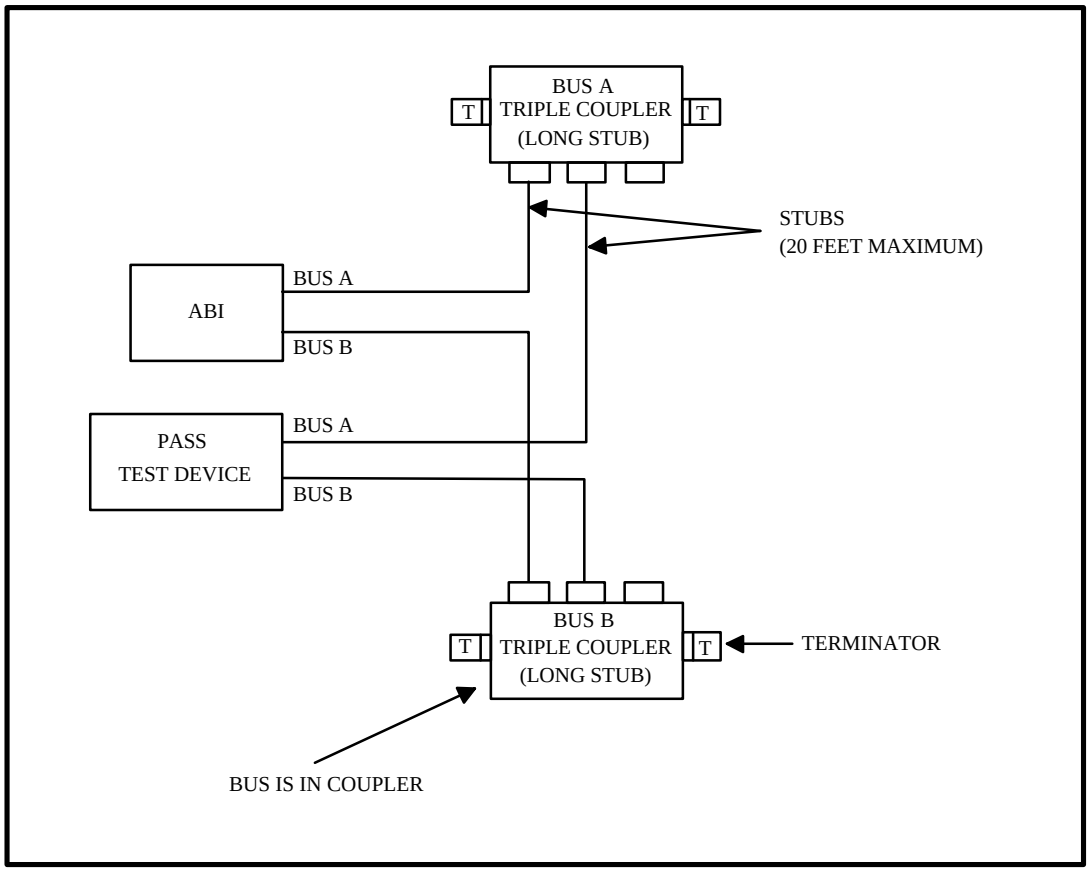


Figure 2-6b <u>Correct</u> Method of Bus Coupling (Long Stubs)

Figure 2-7 shows a short-stub connection. The total stub length must be less than one foot. Since this includes the length of the stub on the board, the actual stub should be less than nine inches. As previously mentioned, this method of coupling is rarely used.

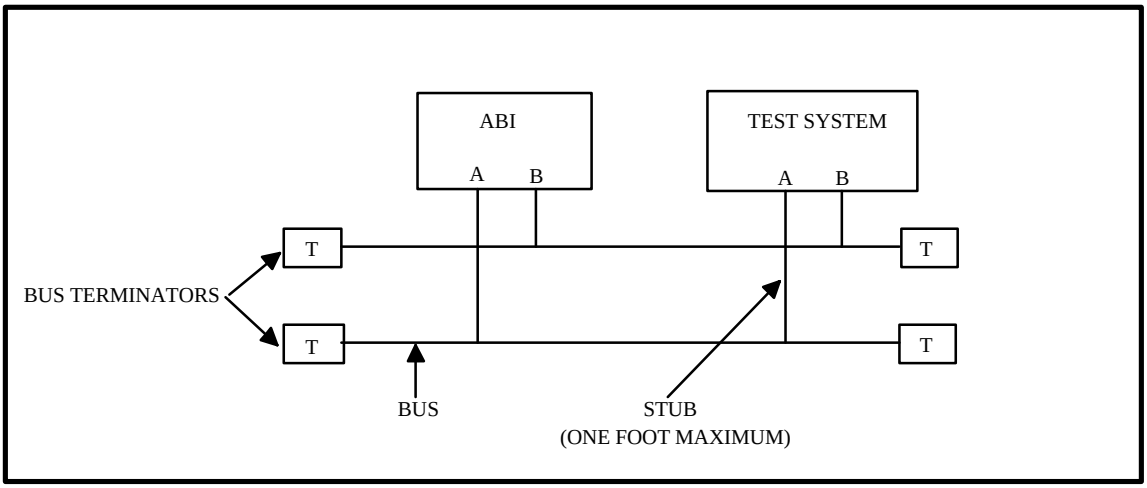


Figure 2-7 Bus Coupling (Short Stubs) - NOT RECOMMENDED

Selecting Long or Short Stub Connection Circuitry

The ABI can provide either a long or short stub interface to a MIL-STD-1553 bus. Jumper block JB1 on the Transceiver Daughter Board is used to select the proper interface circuitry on the board. The board is factory jumpered for long stub connections, which is by far the most common method used. If you are not sure which configuration is used with your system, set the board for long stub coupling. The jumpers used to make these selections are shown in Figure 2-8. If you wish to verify or change these jumpers, they should be set as shown in the figure.

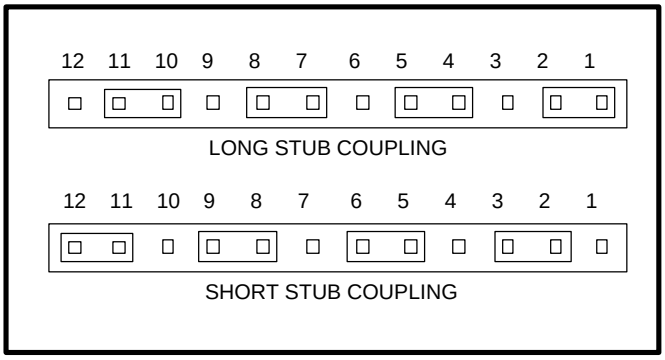


Figure 2-8 Long or Short Stub Selection with JB7 and JB8

Power Application

The next step is to power-up the computer system. Figure 2-1 shows two LEDs that are visible on the board's rear panel. These LEDs are labeled 1 and 2. LED 2 indicates proper initialization of the board at power-on time. LED 1 indicates that the board has been enabled using bit 0 of the I/O Control Register. At power-up, LED 2 should be illuminated and LED 1 should not be illuminated. If this is not the case, contact SBS Engineering.

Testing the Board

Table 2-7 contains the bit definitions for the Control Register, a read/write eight-bit register. This register must be accessed, and bit 0 written to a "1" before the board can be accessed. When bit 0 is a "1", LED 1 will illuminate.

Table 2-7 Control Register Bit Definitions

Bit Position	Name	Definition
0	Enable	Board Enable. When written as a "1" the board is enabled. A "0" is used to disable the board. After power-up the board is disabled.
1	Mode	Block Size Mode. This bit selects between a 128K byte buffer length ("0"), and a 64K byte buffer length ("1").
2	Select	Block Select. This bit is only valid when 64K byte block mode is selected. When this bit is a "0" the lower 64K bytes of ABI memory are selected for accessing. When this bit is "1" the upper 64K bytes of ABI memory are selected.

When both LEDs are illuminated, the board can be accessed from the host at the address selected by the user during configuration. The user must determine a suitable method for accessing the board from software. The method used varies depending on the language used for programming.

Once the method has been determined, the user must write a program to test the board. SBS Engineering supplies a number of diagnostic and utility functions on the board in microcode which may be used to test the operation of the board. The section entitled "Built-In Test Routines" discusses a recommended procedure to be followed at this point to test the board. This test procedure should normally be included in a daily operational readiness user program.

Variable Voltage Control of the 1553 Bus Waveforms

A voltage reference controls the peak-to-peak voltage level of the 1553 bus signals. The user can adjust this voltage reference by reprogramming a digital-to-analog converter (DAC). After power-up, the DAC is initialized to a setting for normal 1553 bus operation.

The DAC output value can be changed by writing a data value to the ABI at offset 1FBI at offset 1Fh. Each data bit has a final reference weighting of 50 millivolts. Normal 1553 bus operation occurs when a value of 00F0h is written to the DAC. Table 2-8 is included as reference only and, due to the loading of the bus, may vary from system to system.

Table 2-8 Variable Voltage Output Selection

DAC Value	1553 Bus Voltage (Peak to Peak)
00F0	21.5
00E0	20
00D0	19
00C0	17.5
00B0	15.5
00A0	14
90	12.5
80	10.5
70	9
60	7.5
50	6.5
40	4.5
30	2.9
20	1.1
10	0.4
0	0

BUILT-IN TEST ROUTINES

Introduction

The ABI contains built-in-test capabilities which provide the user with an added level of hardware confidence and help to ease integration into new or existing systems. Many different routines are available to provide the user with information about various subsystems or devices on the ABI. This section describes each of these test routines, demonstrates how to execute them and explains how to interpret the results. Troubleshooting tips are also provided to assist in the initial integration and subsequent use of the board.

Microcode Organization

The ABI microcode is stored in a set of five registered PROMs with a capacity of 8K bytes of microprogram data. The address bus for these PROMs is a 12-bit pipelined bus. An additional bit is used to select a high or low PROM area (See Table 4-2 in the "ABI Operations Overview" subsection of the *Full Function Programmer's Reference*). This **prom select** bit allows the microcode PROMs to store two separate and independent microprograms. When this bit is set to "0", the low PROM area is selected. When the thirteenth bit is set to "1", the high PROM area is selected.

Each PROM area (high and low) contains Built-In-Test (BIT) microcode followed by application microcode. The ABI may select a specific BIT routine or MIL-STD-1553 application microcode by calling one of 15 starting addresses. These microprogram starting addresses are listed in Table 3-1.

Table 3-1 BIT Addresses

Microprogram Address	Program Selected	Description
0001h	Built-In-Test # 1	Tests Internal registers and addressing modes.
0002h	Built-In-Test # 2	Tests ALU and condition code operation.
0003h	Built-In-Test # 3	Tests RAM with Read/Write/Verify pattern-specific data
0004h	Built-In-Test # 4	More testing of internal registers and addressing modes.
0005h	Built-In-Test # 5	Tests on-board timer sub-system.
0006h	Built-In-Test # 6	Tests 1553 interface logic and front end.
0007h	Built-In-Test # 7	Tests external discrete trigger function
000Dh	Application Code	Starts application code. (See "Microcode Init" subsection of *Full Function Programmer's Reference*)

The following pages show how to perform each BIT and how to interpret the results.

NOTE FOR ABI-PC2 USERS:

Currently, the Built-in Test Routines included with the ABI-PC2 interface board are not all operational. Please note the following:

ABI-PC2, Normal Configuration
Do not run BIT 7 or the "ALL BIT" test (which includes BIT 7). BIT 7 does not run properly and will result in a system "lock-up".

ABI-PC2, PASS-1000 Configuration
Do not run BITs 5, 6, 7 or the "ALL BIT" test.

In addition, do not use the "RUN ALL BIT" test included with the Interface Libraries.

Executing the ABI Built-In-Tests

Certain memory offsets must be loaded with specific values prior to executing a built-in test. Instructions for programming these locations are included with each test procedure.

BIT # 1

Procedure:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)

2. Set MAP register by loading offset 0010h with "1". (Set MAP)

3. Load offset 003Fh with any value.

4. Load offsets 0040h to 004Fh with "0".

5. Load offsets 0044h, 004Ch, and 004Eh with "FFFF".

6. Start ABI by loading offset 0000h with "0042". (Start)

BIT #1 is complete when offset 0040h has changed from "0" (after a few microseconds) to a non-zero value. Check the results against Table 3-2.

Results:

Table 3-2 BIT #1 Results		
Address	**Expected Contents**	**Description**
0040h	non-zero value	BIT Code version number. Exact number may vary, but should be similar to 0402.
0041h	0001	Base register 0, pipeline source okay
0042h	0002	Base register 1, pipeline source okay
0043h	FFFF	Indirect register 0 , pipeline source okay
0044h	0000	Indirect register 1, pipeline source okay
0045h	XXXX	Echo input. Value written here will also be written to location 46 when BIT 1 is executing.
0046h	(45)	Echoed value of contents of location 45.
0048h	XXXX	Counter pattern. Indicates microcode is operating. Will be different every time it is read.
0049h	0001	Base register 0, ALU source okay
004Ah	0002	Base register 1, ALU source okay
004Bh	FFFF	Indirect register 0, ALU source okay
004Ch	0000	Indirect register 1, ALU source okay
004Dh	FFFF	Base register 0, ALU Instruction source okay
004Eh	0000	Base register 1, ALU Instruction source okay
004Fh	non-zero value	Revision number for entire diagnostic and utilites microcode. Exact number may vary.

BIT # 2

Procedure:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)
2. Set MAP register by loading offset 0010h with "2". (Set MAP)
3. Load offset 003Fh with any value.
4. Load offset 0040h with "FFFF".
5. Start ABI by loading offset 0000h with "0042". (Start)
6. Wait four microseconds.
7. Check contents of offset 0040h.

Results:

The contents of offset 0040h should be "0". Any bits set to "1" indicate an error; these are explained in Table 3-3.

Table 3-3 BIT #2 Errors

BIT #	INDICATES
1	Failure of ZERO Condition Code
2	Failure of NEGATE Condition Code
3	Failure of CARRY Condition Code
4	Failure of OVERFLOW Condition Code

BIT # 3

Procedure:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)

2. Set MAP register by loading offset 0010h with "3". (Set MAP)

3. Load offset 003Fh with any value.

4. Load offset 0040h with "FFFF".

5. Start ABI by loading offset 0000h with "0042". (Start)

6. Load **starting** test address in offset 0044h.

7. Load **ending** test address in offset 0045h.

8. Start the test by writing one of the following values to the Command (CMD) word at offset 0040h.

Value	Result
"1"	Start test and perform one time only from the starting address to the ending address.
"2"	Start test and perform continuously from the starting address to the ending address. Halt only if an error occurs.
"3"	Start test and perform continuously from the starting address to the ending address. Do not halt even if an error occurs.

BIT #3 performs a series of READ/WRITE/VERIFY operations, beginning with the starting address and continuing to each subsequent address. There are three values, "0000", "FFFF" and DATA=ADDR, used for these operations. The operations are performed for a random number of address locations using one of these values (i.e. "0000"). The BIT test then uses another of these values (i.e. "FFFF") to test the next group of addresses. After a random number of addresses have been tested with the second value, the BIT test will begin using the third value for the operations. This test pattern continues through the ending address. If more than one cycle has been requested, the next pattern will be different from the previous. The address locations used by the BIT #3 microcode are listed in Table 3-4.

Results:

Table 3-4 BIT #3 Addresses	
Address	**Description**
40	CMD - Command word, used to start test.
41	RSP - Response word. Microcode copies the CMD word to this location.
42	LCNT - Loop Count. Number of test loops completed.
43	EADR - Error Address. Indicates address where R/W/Verify error.
44	LOW - Starting Address for test range.
45	HI - Ending Address for test range.

Built-In Test # 4

Procedure:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)
2. Set MAP register by loading offset 0010h with "4". (Set MAP)
3. Load offset 003Fh with any value.
4. Load offsets 0040h through 0042h with "0000".
5. Start ABI by loading offset 0000h with "0042". (Start)
6. Wait for one second, or for offset 0040h = "FFFF".

Results:

The information shown in Table 3-5 is provided by the microcode.

Table 3-5 BIT#4 Results

Address	Contents	Description
0040h	FFFF	Indicates test completion.
0041h	0000	No Errors.
0041h	0001	Failure during Base Register 0 Verify.
0041h	0002	Failure during Base Register 1 Verify.
0042h	xxxx	Base register contents at time of failure.
0043h	xxxx	Desired value of data.
0044h	xxxx	Actual value of data.

This test performs Base Register tests similar to those in BIT # 1.

Built-In Test # 5

Procedure:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)
2. Set MAP register by loading offset 0010h with "5". (Set MAP)
3. Load offset 003Fh with any value.
4. Load offsets 0040h through 0044h with "0000".
5. Start ABI by loading offset 0000h with "0042". (Start)
6. Wait for one second, or for offset 0040h = "FFFF".

Results:

The information shown in Table 3-6 is provided by the microcode.

Table 3-6 BIT #5 Results

Address	Contents	Description
40	FFFF	Indicates test completion.
41	0000	Error Code, 0000 = No Error.
"	0001	Timer Write/Read Failure.
"	0002	Timer Count/Read Failure.
"	0003	OVFL Discrete will not reset.
"	0004	DOVFL Discrete will not reset.
"	0005	OVFL Discrete set prematurely.
"	0006	DOVFL Discrete set prematurely.
"	0007	OVFL does not set.
"	0008	DOVFL does not set.
42	xxxx	Desired Value of Data.
43	xxxx	Actual value of Data.

Built-In Test # 6

Procedure:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)
2. Set MAP register by loading offset 0010h with "6". (Set MAP)
3. Load offset 003Fh with any value.
4. Load offsets 0040h through 005Fh with "0000".
5. Load offset 001Fh with "00F0". (Set variable voltage control register.)
6. Start ABI by loading offset 0000h with "0042". (Start)

NOTE: This test must be performed on a **properly terminated** 1553 bus or with terminators installed directly on the front panel connectors. The ABI is intended to be connected to the 1553 bus via a data bus coupler if long stub operation is selected. There must be no extraneous activity on the bus while BIT # 6 is being performed. If BIT #6 fails while the ABI is connected to the 1553 bus: disconnect the ABI from the bus, install terminators directly onto the board and perform the test again.

Results:

The information shown in Table 3-7 is provided by the microcode.

Table 3-7 BIT #6 Addresses

Address	Contents	Description
0040h	CMD	Command Word. Should be "0000".
0041h	LPC	Loop count.
0042h	ERC	Error counter.
0043h	TNM	Test number of failure.
0044h	ERR	Error number.

If the bus is properly terminated and BIT #6 fails, note the TNM and ERR values and contact SBS Engineering. These values are defined in Tables 3-8 and 3-9.

Table 3-8 TNM Values	
TNM Value (in hex)	**Description**
0010	Prime Bus Active (PBA)/Alternate Bus Active (ABA) Initial Check.
0021	Serial Data Transmit/Receive Test.
0022	Serial Data Transmit/Receive Test.
0023	Serial Data Transmit/Receive Test.
0024	Serial Data Transmit/Receive Test.
0031	Gap Timer Test # 1.
0032	Gap Timer Test # 2.
0033	Gap Timer Test # 3.
0034	Gap Timer Test # 4.
0035	Gap Timer Test # 5.
0036	Gap Timer Test # 6.
0041	Data Lost Bit Test.
0042	Data Lost Bit Test

Table 3-9 ERR Values	
ERR Value (in hex)	**Description**
0001	PBA = 1 > 20 Microseconds Error.
0002	ABA = 1 Error.
0003	Remote Terminal/Subaddress Data Error.
0004	Data Error.
0005	Invalid Word Error.
0006	Sync Type Error.
0007	Gap Too Short Error.
0008	Gap Too Long Error.
0009	Gap Static Data Error.
000A	Gap Counter Data Error.
000B	Lost Not Set Error.
000C	Lost Set 1 Error.
000D	Lost Set 2 Error.
000E	PBA Time-out Error.
000F	Data Ready (DRDY) Time-out Error.
0010	SARDY Time-out Error.
0011	XRDY Time-out Error.

Built-In Test # 7

Procedure:

1. Reset ABI by loading offset 0000h with "1". (Reset CSR)
2. Set MAP register by loading offset 0010h with "7". (Set MAP)
3. Load 003Fh with any value.
4. Load offsets 0040h through 0043h with "0000".
5. Start ABI by loading offset 0000h with "0042". (Start)
6. Load offset 0040h with a data value of "0001".
7. Verify that offset 0041h = "0001".
8. Repeat steps 5-7 using a data value of "0000".
9. Each time offset 0040h is loaded with "0000" or "0001", an echo of that value will appear at offset 0041. BIT #7 will continue to check registers 0040h and 0041h indefinitely until the ABI is reset. Steps 5-8 may be repeated while BIT # 7 is executing.

NOTE: This test must be performed with a jumper or shunt installed on the front panel between the Discrete Input and Discrete Output, pins P7-9 and P7-10. Refer to the *Hardware Installation* section for details on the location of these pins. This test has been included for future capabilities of the ABI. The discrete I/O functions are not used at this time. Execution of BIT 7 is optional and is left to the user's discretion.

FULL FUNCTION PROGRAMMER'S REFERENCE

The Advanced Bus Interface (ABI) module provides the host computer with a real-time MIL-STD-1553 interface. The ABI is capable of simultaneous and independent Bus Controller (BC) and 31 Remote Terminal (RT) simulation as well as Map and Sequential Monitoring operations. This section of the manual details the operations of the ABI and contains the following subsections:

- Introduction / ABI Operations Overview
- Microcode Initialization
- BC Simulation
- RT Simulation
- Bus Monitoring
- Interrupts

NOTE: This manual section assumes the user has previous knowledge of MIL-STD-1553 operations.

The "ABI Operations Overview" subsection discusses ABI memory organization and microcode program operations. The "Microcode Initialization" subsection explains the required steps in initializing an ABI. The three subsections following Microcode Initialization detail ABI configuration for Bus Controller, Remote Terminal and Monitor operations. The "Interrupts" subsection explains ABI interrupt capabilities and the recommended servicing procedures for hardware and software polled interrupts.

The ABI may have two 1553 microcode protocols resident on-board. (See the cover sheet of this manual for the available microcode loads.) The user may switch between these protocols through a software bit switch in a key control register. This section details the "Full Function 1553B" protocol. Please see the "Microcode Initialization" subsection for more details on selecting and initializing the desired protocol. Contact SBS Engineering for more information on special protocol requirements.

The accompanying manual, 1553 Interface Libraries, provides a detailed description of the software libraries provided by SBS. These libraries provide the interface to a customer's real-time application software and are compatible with most computer operating systems. Review this *Full Function Programmer's Reference* section prior to reading the 1553 Interface Libraries manual.

ABI OPERATIONS OVERVIEW

The ABI is a bit-slice microcomputer with 8Kx40 bit microcode instructions containing programs customized for real-time 1553 applications. Figure 4-1 provides a block diagram of the ABI. When installed in the host computer, the ABI appears as a contiguous block of virtual or physical 64Kx16 bit memory (128K bytes). This pseudo dual-port memory resides on the ABI and is mapped or connected to the host system for user application program access. The ABI module provides an arbiter circuit to allow access to memory by either the on-board microcomputer or the host system on request.

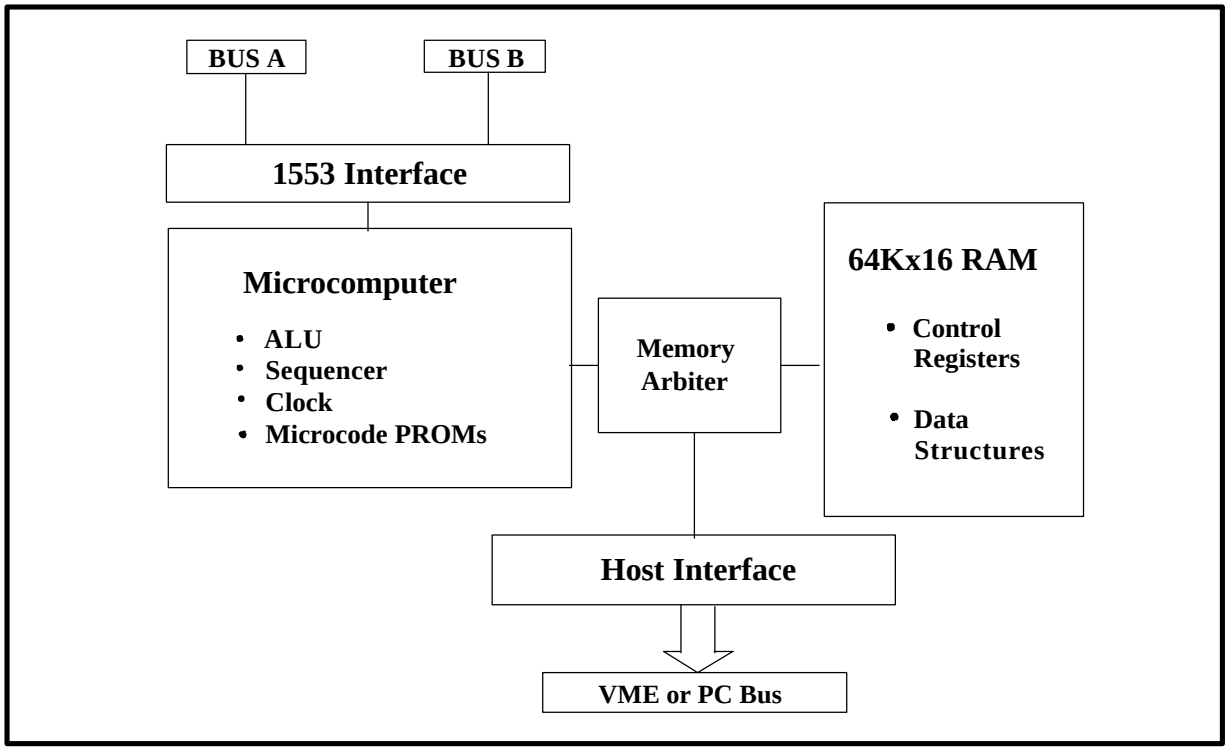


Figure 4-1 ABI Architecture

The user's controlling application program initializes and downloads (into the ABI memory) the data structures and control registers for BC, RT, Monitoring and Interrupts. BC data structures consist of linked-list encoded command blocks which the microcode program interprets for proper BC command message simulation. The BC linked-list allows for easy construction of minor and major frame formats. RT data structures consist of word array blocks or tables, arranged by RT address, which provide holding areas for Status and Data Words. There are two types of monitor structures: Map Monitoring utilizes linked-list word buffers and Sequential Monitoring uses double word buffers. Map Monitoring stores real-time 1553 message traffic in linked-list data word buffers shared with RT functions. Sequential Monitoring provides double-buffering for time-stamped 1553 data logging. For interrupt events there is a double-buffered Interrupt queue data structure which provides the host system

with information for ABI generated interrupts. All of these data structures have root addresses and setup information stored in a small area of reserved RAM called **control register** memory.

The host system always has real-time access to the data buffers used for BC command blocks, RT simulation and map monitoring. It also has real-time access to sequential monitor buffers and interrupt queues. This access provides the user with an advanced 1553 interface to the host system.

The following paragraphs detail the host interface to the ABI control registers and 1553 data structures.

Addressing Considerations for Host Application Programs

The ABI module is a shared memory device where the 64Kx16 memory appears to the host system software as a single contiguous block. Virtual memory systems such as UNIX and VMS require some type of mapping procedure to tie the physical ABI memory address to the actual host program memory. Physical memory systems such as VxWorks, DOS and PDOS can usually access the ABI memory directly through program addressing. Please see the 1553 Interface Libraries manual for more details on ABI module mapping to host computer systems.

NOTE: The discussions throughout this subsection refer to word (16 bits) addressing of ABI control registers and data structures. Like the 1553 bus, the ABI interface is 16-bit word aligned with the right-most bit being the least significant and numbered "0". Most host systems access memory on byte boundaries which means any address mentioned in this section must be multiplied by two (or shifted one bit left) to determine the proper byte address. Tables 4-1 and 4-4 list user accessible control registers and provide both the word and byte address in the ABI memory space.

When an ABI is installed, a unique base address for the board must be selected in the host address space. This manual section refers to offsets relative to a base address of zero and in **word** boundaries (addresses 0000-FFFFh). Host systems which simply use physical addressing to access the board construct an address by adding the offset to the base address. If this type of host system places the ABI module at **byte** address 200000h, the ABI byte address range will be 200000h to 21FFFEh (FFFFh is shifted one bit left to obtain 1FFFEh, then added to base address 200000h).

Virtual memory systems obtain a pointer from the operating system which is used to subsequently access the board. This pointer is used with the offset to access a specific location on the ABI. *Note that the ABI does not support byte accesses.*

ABI Memory Organization

Figure 4-2 shows the RAM sections for ABI control registers and data structure area. The ABI is unique in that user defined BC command block chains, RT response tables, Monitor data structures and Interrupt queues can be located in any location above 03FFh. This allows great flexibility in the host software because data structures can be defined in host memory and copied into any ABI memory location above 03FFh. Areas below 0400h are reserved for control registers that the host software accesses for hardware initialization and pointer definitions to data structures in the 0400h to FFFFh area.

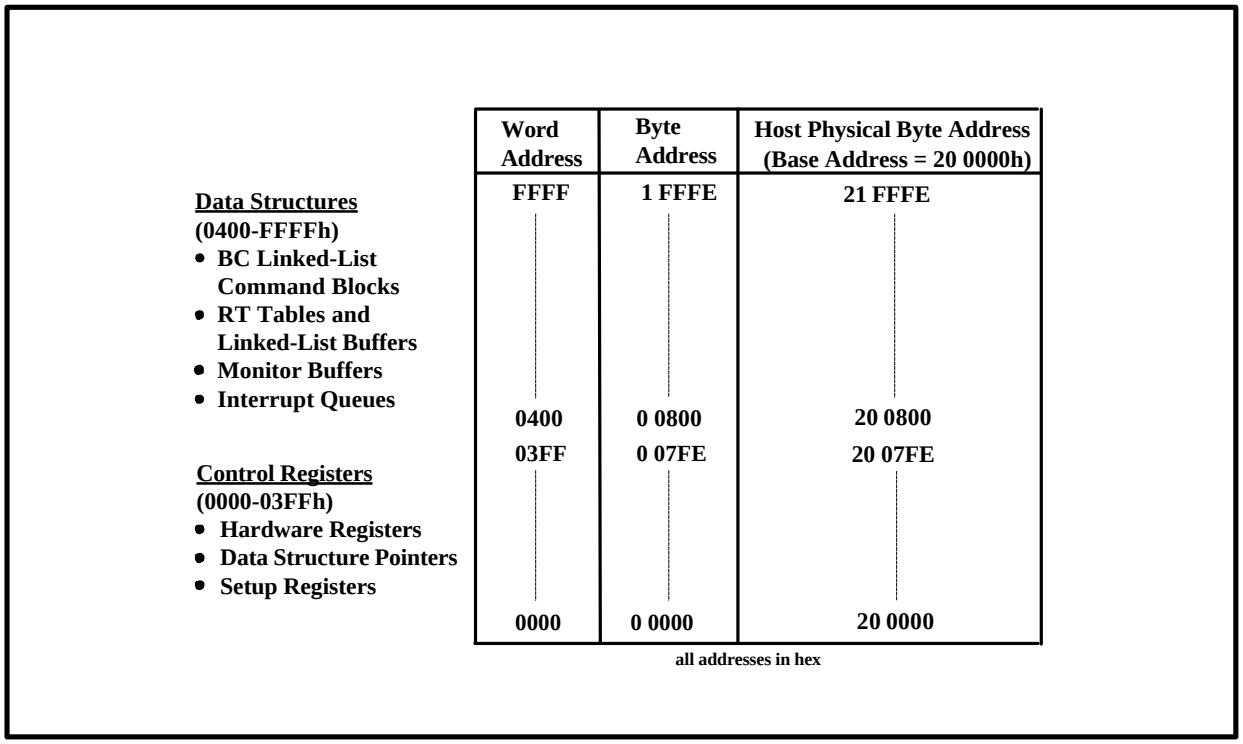


Figure 4-2 ABI Memory Organization

Upon ABI power-up, the host software performs two steps to process 1553 data.

1. Initializes ABI memory.

2. Starts up an on-board microcode program through a defined procedure of accesses to control registers.

Next, the host software loads BC, RT, Monitor and Interrupt data structures into upper ABI memory, and programs the control registers in lower ABI memory with initial data structure addresses. The host then sets a flag in a reserved control register that tells the ABI microcode that data structure addresses have been loaded and 1553 data processing can begin.

The following paragraphs review the control register area of memory and provide useful hints for successful ABI operations. The subsections "BC Simulation", "RT Simulation", "Bus Monitoring," and "Interrupts" detail the actual data structures used in each mode of operation.

Data Structure Memory: 0400-FFFFh

This area of memory is reserved for user configured data structures for BC command lists, RT response and data buffers, Monitor buffers and Interrupt queues. All data structures have root addresses held in a respective control register as detailed in Table 4-1. Details of data structures for BC, RT, Monitor and Interrupt functions are detailed in their respective subsections of this manual.

There are required minimum length data structures for RT Simulation/Map Monitoring, Sequential Monitoring, and Interrupt queues. Required structures occupy only a fraction of the data structure area (less than 512 words), leaving nearly 63 kilobytes for user defined applications.

Control Register Memory: 0000-03FFh

The memory area below 0400h is reserved for ABI hardware control registers, data structure pointers and setup registers. The first 64 words (0000 to 003Fh) are ABI hardware registers which are accessed by the host system only during ABI startup, ABI clock programming/reading and for interrupt handling. The main hardware register is the Control/Status Register (CSR) located at address 0000. This register enables ABI microcode operations and controls other fundamental operations like interrupt enabling. Besides the CSR, there are only a handful of hardware registers the host software should access (for proper microcode program startup and setting ABI clock options).

Bit 8 (Disable I/O) in the CSR enables or disables access to the hardware registers (the host always has access to the CSR register). The "Microcode Initialization" subsection of this manual details the proper startup procedure and the disabling of hardware register access.

Control Registers 0040h - 03FFh contain microcode program registers, pointers and setup registers and microcode scratch-pad registers. The host software uses this area of memory to initialize addresses and buffer sizes for data structures located above 03FFh. Some control registers in this area provide definition settings for 1553 simulation and monitoring functions, such as BC intermessage gap times, BC retry counts, RT response times, etc. Another group of control registers in this area include an error table which provides the host with a comprehensive listing of the error status of 1553 protocol and ABI microcode execution. Tables 4-1 and 4-4 provide a detailed description of all user accessible control registers from 0000h to 03FFh. **Reading or writing to control register locations not listed in these tables will produce unpredictable results.**

Control Register Descriptions

Tables 4-1 and 4-4 provide a comprehensive list of all user-accessed control registers in the 0000 to 03FFh memory area. Table 4-1 is divided into functional sections. Each register has a program acronym, word and byte address, page reference (in this manual) and a description. An asterisk ("*") after an acronym indicates this register must be programmed with a value or an address to a required minimum length BC, RT, Monitor or Interrupt data structure. Hardware registers are discussed following the table.

Table 4-1 ABI CONTROL REGISTERS

Acronym (host access)	Word Address Byte Address (in hex)	Page Ref.	Description
HARDWARE / CONTROL			
CSR (r/w)	00 000	4-10	Control/Status Register. Key register for proper initialization and operation of ABI. This is the only register always accessible to the host. See the description following this table.
MAP (wo)	10 020	4-10	Holds microcode starting address. See the description following this table.
GINT (wo)	1A 034	--	Use to test hardware interrupt handshaking. Write any value to this register to generate an interrupt. (Does not affect the interrupt queue.)
VVDR (wo)	1F 03E	4-14	Controls output voltage. See the description following this table.
PSTEP (wo)	3F 07E	4-10	Causes "pipestep" instruction to occur. See the description following this table.
MICPC (ro)	40 080	4-21	Microcode Product Code
MICVC (ro)	41 082	4-21	Microcode Version Code
CMD (wo)	80 100	--	Commands the starting of 1553 bus processing. Write a non-zero value to this register to start bus processing. Write a zero to this register to halt bus processing. Program registers 82h to 9Ch prior to setting this register to a non-zero (active) value.
RESP (ro)	81 102	--	This word continuously increments in RESPONSE to CMD (above) not set to zero. This provides a simple check that the micromachine is running. While counting, this word should never be equal to zero (zero is skipped in the counting).
SYSTEM CLOCK			
SCHIGH (r/w)	93 126	4-11	The microcode uses an internal 32-bit, 32-microsecond timer. SCHIGH is the most significant 16 bits of this timer.
SCLOW (r/w)	94 128	4-11	SCLOW is the least significant 16 bits of the system timer.

Table 4-1 ABI CONTROL REGISTERS			
Acronym (host access)	Word Address Byte Address (in hex)	Page Ref.	Description
CCW (r/w)	9D 13A	4-11	Clock Control Word. Each bit corresponds to an action. Bit 0: read timer and update SCHIGH and SCLOW, interrupt code 1 when update is valid; Bit 1: read timer and update SCHIGH and SCLOW, no interrupt; Bit 2: Reset internal timer with values stored in SCHIGH and SCLOW. Updates and resets are done within 100 microseconds and only while the ABI micromachine is running (CMD is non-zero).
ITDRL (wo)	31 062	4-13	Lower 16 bits of timer. DO NOT READ FROM THIS REGISTER. See the description following this table.
ITDRH (wo)	33 066	4-13	Upper 16 bits of timer. DO NOT READ FROM THIS REGISTER. See the description following this table.
ITL/ HDAT0 (ro)	35 06A	4-13	Dual function register: when written to (by microcode only) = timer latch / when read from = lower 16 bits of system timer. DO NOT WRITE TO THIS REGISTER. See the description following this table.
ITCR/ HDAT1 (r/w)	37 06E	4-13	Dual function register: when written to = Internal Timer Control Register / when read from = upper 16 bits of system timer. See the description following this table.
INTERRUPT PROCESSING			
IQRSP (wo)	82 104	4-60	Interrupt Queue Response flag word: If IQRSP=FFFFh, Host is processing interrupts; if IQRSP=0001h, processing is complete.
IQPTR1 * (r/w)	83 106	4-60	Offset to interrupt queue 1. This queue is processed by the Host computer. Each entry consists of four words: Word 1 is the interrupt code; Words 2-4 depend on the interrupt code.
IQPTR2 * (r/w)	84 108	4-60	Offset to interrupt queue 2. This is the queue which is currently active. The Host should NOT process this queue.
IQCNT1 (ro)	85 10A	4-60	The number of entries in interrupt queue 1. This is used by the Host computer's ISR to process the IQPTR1 queue.
IQCNT2 (wo)	86 10C	4-60	The number of entries in interrupt queue 2. **Do not access this register while the ABI is processing 1553 data.**
IQNUM (r/w)	87 10E	4-60	The number of interrupt slots in each queue. The default is 4 entries. If the number of entries exceeds the value in IQNUM, an error is generated.
SEQUENTIAL MONITOR			
M1PTR * (wo)	8C 118	4-50	Offset to sequential monitor buffer 1. This is the active buffer. **Do not access this register while the ABI is processing 1553 data.**
M2PTR * (r/w)	8D 11A	4-50	Offset to sequential monitor buffer 2. This is the non-active buffer. This buffer should be processed by the Host computer after M1PTR and M2PTR are swapped. Wait a minimum of 20 microseconds after a swap before processing the M2PTR table.
MBLEN * (wo)	8E 11C	4-50	Unsigned 16-bit number indicating length in words of each sequential monitor buffer. Minimum length is 50 words. The microcode will swap buffer pointers when the number of words left in the buffer is less than the minimum buffer length.

Table 4-1 ABI CONTROL REGISTERS			
Acronym (host access)	Word Address Byte Address (in hex)	Page Ref.	Description
MBFLG (r/w)	9F 13E	--	Sequential monitor buffer control word. Set to "1" to cause the monitor buffer pointers to swap. Set to "2" to cause a swap and a Monitor Buffer Swap interrupt code 2. When the operation is completed, the microcode sets this word to "0".
SMBCNT (wo)	AB 14D	4-51	Sequential Monitor Buffer Counter.
BUS CONTROLLER (BC)			
BCIPTR (wo)	8F 11E	4-26	Initial Bus Control program block. This location is continuously monitored by the ABI's microcode. When BCIPTR is non-zero, the microcode will execute a chain of BC program blocks beginning at the offset defined in BCIPTR.
BCCPTR (r/w)	90 120	4-26	Contains the offset to the currently active BC program block. If the LINK word of the block is set to zero, BC execution will be halted upon completion of this block.
BCLPTR (r/w)	91 122	4-26	Contains the offset to the last BC program block that executed before the microcode stopped processing BC program blocks.
BCSMSK (r/w)	92 124	4-27	BC status mask. If bit 2 of the flag word in a BC program block is set to "1", the contents of BCSMSK is logically "ANDed" with the returned status word(s). If the resultant is non-zero, an interrupt occurs. The Continue-On-Error/Masked Status bit in the flag word of the BC program block determines if the BC chain will be stopped.
BCERVL (ro)	A7 14E	--	Contains the last BC error table entry. This value is reset to "0" at the start of each BC chain.
BCIGP (r/w)	AE 15C	4-48	Use to change the intermessage gap time for BC messages: *BCIGP = (time_in_microseconds - 4) * gap_count* (time_in_microseconds = desired gap time, must be > 4) (gap_count = MIP rating of ABI) Accurate to within five microseconds of desired gap time.
BC RETRY			
BRTCNT (r/w)	AF 15E	4-27	Number of retries to be performed for a BC error condition. The maximum is 16.
BRTBUS (r/w)	B0 160	4-27	Defines bus (A or B) to be used for BC retries. Bit 0=1st retry of message, Bit 1=2nd retry of message, etc. If bit=0, retry is on the same bus as original message. If bit=1, retry is on the opposite bus.
BRTCMD (r/w)	B1 162	4-27	Contains the last BC command that was retried. This value is reset to "0" at the start of each BC chain.
BRTRTC (r/w)	B2 164	4-27	Contains the actual number of retried messages. This value is reset to "0" at the start of each BC chain.

Table 4-1 ABI CONTROL REGISTERS			
Acronym (host access)	Word Address Byte Address (in hex)	Page Ref.	Description
REMOTE TERMINAL (RT)			
SWTPTR * (r/w)	88 110	4-30	The Status Response Table pointer is an offset to the beginning of a 64-word table. The first 32 words are for the Host computer; the last 32 words are for the microcode. Each entry corresponds to an RT. The actual status response word sent to the 1553 bus is obtained through a logical "OR" operation involving the Host entry and the microcode entry for a particular RT, except for the ME bit. The ME bit is set by the microcode in the microcode entry. A non-zero entry enables the corresponding RT for simulation. RT 0 is a special case; it is enabled by storing "0400h" in the first entry of the table.
ATPTR * (r/w)	89 112	4-32	The Address Table pointer is an offset to a 32-word table arranged by RT (the RT address table) whose entries are each an offset to a buffer pointer table. A buffer pointer table is a table of 64 words; the first 32 are for the receive subaddresses, the last 32 are for the transmit subaddresses. Each entry is an offset to an RT data buffer linked list. An RT data buffer consists of a link offset, a flag word, and 32 data words.
FTPTR * (r/w)	8A 114	4-35	Filter Table pointer is an offset to a table of 32 words arranged by RT number. Each entry of this table is an offset to a subaddress filter table. A subaddress filter table is 64 words: 32 for receive subaddresses, 32 for transmit subaddresses. Each entry of the subaddress filter table contains control bits for a particular RT-t/r-SA. Bit 0: Sequential Monitor Enable, Bit 1: Interrupt on Message Received, Bit 2: Sequential Monitor Buffer Swap on Message, Bit 3: Generate Parity Error.
RT MODE COMMANDS			
RTPPTR * (r/w)	95 12A	4-41	Remote Terminal Phase Table is an offset to a 32-word table arranged by RT number. It is used by the microcode for saving current transmitter state in response to Transmitter Shutdown mode code and for saving current terminal flag inhibit state for Inhibit Terminal Flag Bit mode code.
BITPTR * (r/w)	96 12C	4-45	BITPTR is an offset to a 32-word table arranged by RT number. This table contains the BIT entry sent for each RT upon receipt of a transmit BIT mode command.
LCDPTR * (r/w)	97 12E	4-44	LCDPTR is an offset to a 32-word table arranged by RT number. This table contains the last command word sent to each RT. The last command for a particular RT is obtained from this table and transmitted upon receipt of a Transmit Last Command mode code.
TVWPTR * (r/w)	98 130	4-42	TVWPTR is an offset to a 32-word table arranged by RT number. This table contains the vector word entry sent for each RT upon receipt of a transmit vector word mode command.

Table 4-1 ABI CONTROL REGISTERS			
Acronym (host access)	Word Address Byte Address (in hex)	Page Ref.	Description
LSWPTR * (r/w)	99 132	4-40	LSWPTR is an offset to a 32-word table arranged by RT number. This table contains the status word for the last valid command for each RT. The last command for a particular RT is obtained from this table and transmitted upon receipt of a Transmit Last Status mode code.
LSYPTR * (r/w)	9A 134	4-43	LSYPTR is an offset to a 32-word table arranged by RT number. This table contains the value sent by the BC for each RT on the last synchronize with data word mode command.
PROPTR * (r/w)	9C 138	4-37	PROPTR is an offset to a 32-word table arranged by RT number. Each entry contains a protocol word definition - Bit 0: must be "0", Bit 1: enable map monitor, Bit 2: dynamic bus control accept, Bit 3: 1553B treatment of SA31, Bit 4: Bus A failure, Bit 5: Bus B failure.
MIMPTR (r/w)	9E 13C	4-47	MIMPTR is an offset to a 64-word table arranged by RT number. Each entry of the table is two words. The bits of an entry correspond to a particular mode command. A set bit indicates that the ABI will generate a MIM interrupt upon receipt of a mode command (corresponding to the RT associated with that set bit).
RT VARIABLE STATUS RESPONSE GAPS			
RSPGPA (r/w)	AC 158	4-48	Use to change the maximum allowable status response gap time: *RSPGPA = (time_in_microseconds - 2) * gap_count* (time_in_microseconds = desired gap time, must be > 2) (gap_count = MIP rating of ABI) Accurate to within five microseconds of desired gap time.
RSPGPS (r/w)	AD 15A	4-48	Use to change the actual RT status response gap time: *RSPGPS = (time_in_microseconds - 4) * gap_count* Accurate to within five microseconds of desired gap time.
LEGEND: * Entry required for proper microcode operation. r/w = read/write, ro = read only, wo = write only Word offset locations between 80h and 3FFh that are not mentioned here are reserved. Accessing these locations will produce unpredictable results.			

Key Hardware Registers

CSR (Control and Status Register)

This is the only hardware register (memory between 0000 and 003Fh) which is always accessible to the host software. The user can read or write to this register because the RAM arbiter will always have direct access to the internal microcomputer register 0000 (access to registers 0001-003F can be disabled through bit 8 of the CSR). This register is key to proper initialization and operations of the ABI. Figure 4-3 and Table 4-2 detail the bit definitions for

the CSR register. Carefully read the bit descriptions because undesired operations may result from inadvertent reads to and writes from this register.

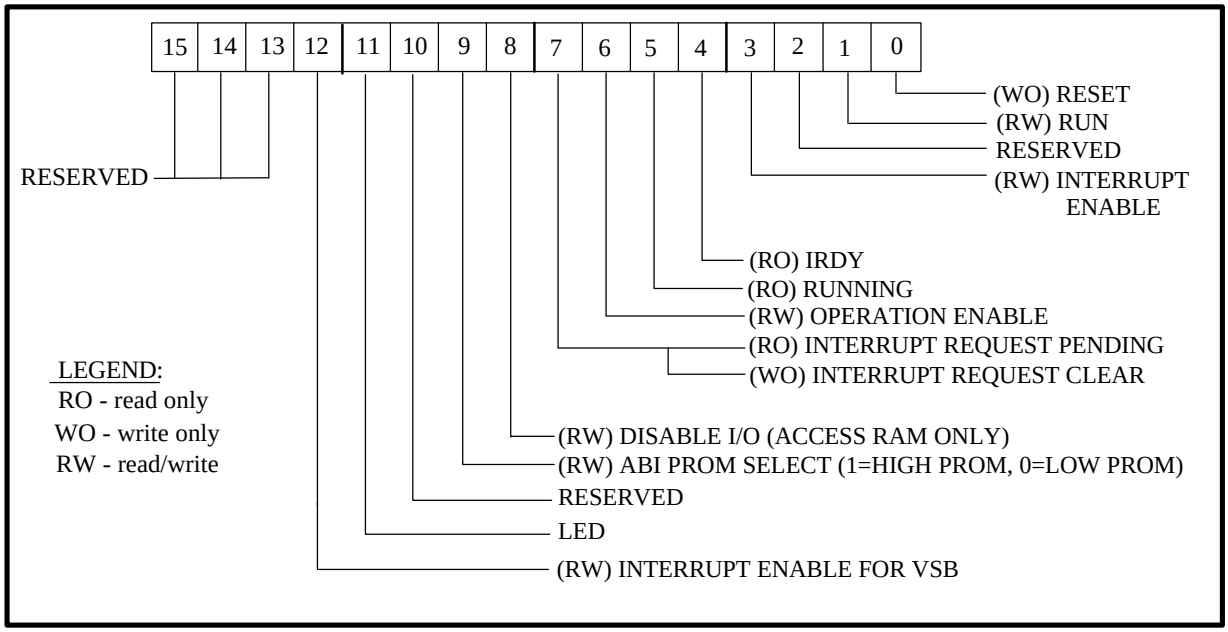


Figure 4-3 Control Status Register Bits

Table 4-2 Control Status Register Bit Definitions

BIT #	NAME	DESCRIPTION
0	RESET	Set this bit to "1" to clear the interface. This will set all CSR bits (including the Reset bit) to "0".
1	RUN	Set this bit to "1" to cause the processor on the ABI to begin execution. Set this bit to "0" to stop all processing.
2	RESERVED	Do not change this bit. (When writing to the CSR, **this bit must be "0"**.)
3	INT ENA	Set this bit to "1" to cause interrupts generated by the ABI to be passed to the host bus.
4	IRDY	This is a status discrete bit which should indicate that the internal ABI logic is ready for processing. It is not used (always low) on current revisions of the ABI-V3.
5	RUNNING	This is a status discrete bit which indicates that the ABI processor is running. When the RUN bit is set to "1" by the host, the ABI microcode will begin execution and will set this bit to "1"; when the RUN bit is set to "0" the microcode will set this bit to "0" .
6	OP ENA	Operation enable is an output discrete bit. When this bit is set to "0", it sets the internal instruction register of the processor to "0". It must be set to "0" prior to restarting a new microprogram and must be set to "1" when the RUN bit is set to "1". **It must not be set to "0" while the ABI processor is running.**

Table 4-2 Control Status Register Bit Definitions		
BIT #	**NAME**	**DESCRIPTION**
7	INT REQ	This bit has two functions: It is read as an Interrupt Request Pending bit and is written to as an Interrupt Clear bit. If this bit is set to "1" when read, an interrupt request is pending. If a "1" is written to this bit, it resets this pending interrupt. **VME board:** This bit and the pending interrupt will be set to "0" by the hardware interrupt acknowledge cycle. **Clear (set to "0") this bit once at initialization and never again unless polling is used.** **PC2 board:** Each time interrupts are serviced, the interrupt handler must clear the pending interrupt.
8	DISABLE I/O	Set this bit to "1" to disable access to the ABI I/O registers 1h through 3Fh. The region 1-3Fh will appear as normal RAM. The CSR is still accessible. It is a good idea to set this bit to "1" after the MAP register has been loaded with the desired microprogram number. **This bit must be "0" to load the map register.**
9	PROM SEL	Set this bit to "1" to select the upper half of PROM for use; set to "0" to select the lower half.
10	RESERVED	Do not change this bit.
11	LED	This bit controls LED 1 on the front panel of the ABI. This bit is active low; set to "0" to turn the LED ON. At power-up or after a reset, this bit is reset to "0" and the LED turns ON.
12	VSB INT ENA	Set this bit to "1" to cause interrupts generated by the ABI to be passed to the host VSB bus.
13-15	RESERVED	Do not change any of these bits.

MAP control register

Write the microcode starting address to this register prior to execution. There are a variety of legal microcode starting address locations that correspond to ABI Built-In Test (BIT) routines, utility microcode programs and the main body of 1553 microcode programs.

PSTEP control register

Write any value to this register to cause a "pipestep" instruction to occur. This instruction latches the microcode and is necessary for proper startup. Access to this register other than for initialization is not recommended.

System Clock Registers **(for ABI-V3 revisions earlier than C, ABI-V2, ABI-PC, and ABI-Q)**

ABI-V3, **revision C or later,** and ABI-PC2 users may access the timing hardware registers in real-time to obtain current time information. The system clock registers are not required for these users.

CCW (Clock Control Word), SCHIGH, SCLOW

Other versions of the ABI require the use of the CCW, SCHIGH and SCLOW registers to obtain the latest time information. For these versions, when the host software needs the latest time information, it sets bit 1 of the clock control word (CCW) to "1". This causes the microcode to calculate the current 32-bit system time and update the SCHIGH and SCLOW registers. When it has updated these registers, the microcode clears CCW to indicate to the host that the time update is complete. The microcode completes this action within 100 microseconds after the host sets bit 1 to "1". The host software may also set bit 0 of the CCW to "1" (instead of bit 1) to cause the microcode to update the time registers and interrupt (interrupt code 1) the host immediately after the time update is complete.

SCHIGH and SCLOW may also be used to preset the 32-bit system time. Load the SCHIGH and SCLOW registers with the desired time and then set bit 2 of the CCW to "1". The microcode will set the system time to these values and then clear the CCW.

Timing Hardware Registers **(for ABI-V3 revisions C and later, and ABI-PC2)**

The **ITDRL**, **ITDRH, HDAT0** and **ITCR** control registers configure the 32 bit hardware timer used for 1553 message time stamping in ABI Sequential Monitoring. This timer may be read, reset or preset as desired by the host software. The ABI timer may be driven from an internal clock or from an external source (front panel or P2 connector). There are jumpers located on the board to configure internal or external time sources and reset lines. Please see the "Hardware Installation" section of this manual for configuration settings. The board defaults to the internal clock setting with internal reset lines.

ITDRL, ITDRH (Internal Timer Data Registers)

Do not read from these registers. These registers are used to pre-load or preset the 32-bit timer used for time stamping 1553 messages. ITDRL is the lower 16 bits and ITDRH is the upper 16 bits of the 32 bit timer. Perform the following steps to write to these registers:

1. Clear bit 0 of the ITCR.

2. Write the desired values to ITDRL and ITDRH.

3. Set bit 0 of the ITCR.

When bit 0 of the ITCR is enabled, the hardware will load the 32 bit internal timer with the programmed values. This function can be performed with internal or external timer configurations.

HDAT0 (Internal Timer Latch)

Do not write to this register. When a value is read from the register, it functions as the **HDAT0** register. HDAT0 provides the lower 16 bits of the system timer and "latches" the upper 16 bits (held in the **ITCR/HDAT1** register). The host software must read HDAT0 before reading HDAT1.

ITCR (Internal Timer Control Register), HDAT1

This control register has a dual function. When a value is written to it, the register functions as the Internal Timer Control Register (**ITCR**), defining the mode of operation of the timer. When a value is read from this register, it functions as **HDAT1**, providing the upper 16 bits of the system timer. This register is located at offset 006Eh and contains three bits, described below as they are used for the **ITCR** function.

Bit 0 - Clock Enable/Disable
Use of this bit is dependent on the settings of JB6, 7, 10 and 11 (See the "Hardware Installation" section of this manual). If the jumpers are set to the default settings of internal clock/internal enable, the internal timer will be enabled when this bit is set to "1". If the jumpers are set for external enable, setting this bit will enable the timer if an external enable signal is present on the external connector. Default for this bit is "1" or enabled.

Bit 1 - Clock Select
This bit selects the clock reference to be used if an internal clock is selected via jumpers. Set this bit to "0" to select a 32-microsecond clock, and set to "1" to select a 1-microsecond clock. If an external clock is selected, this bit is not used. Default is for 32 microsecond resolution.

Bit 2 - Reset Clock
This signal is used to clear the timer from software. To clear the timer to a count of zero, set bit 2 to "1" and hold at "1" for the entire clock period. For example, if a 32 microsecond clock period is selected, bit 2 must be held at "1" for 32 microseconds. The timer will remain cleared until bit 2 is cleared and the timer is enabled.

The timer may also be cleared using external clear signals. These signals must also be held for the duration of the clock period. See the "Hardware Installation" section for more information.

NOTE: The state of the Reset Clock or Clock Select bit should only be changed when the timer is disabled.

VVDR (Variable Voltage DAC Register)

This register (offset 001Fh) controls the amplitude of ABI transmitted 1553 signals (during BC or RT simulation). By writing to this register, the host software can set the ABI output voltage levels from 0 to 21.5V (peak-to-peak). Table 4-3 provides typical output voltages for given register values. The actual output voltage may be affected by bus loading (the number of terminals, length of bus, etc.). The microcode 1553 main body initializes this register to 00F0h (about 21.5 volts). The BIT 6 routine and Send Utility microcode programs require this register to be programmed for proper operations.

Table 4-3 Variable Voltage DAC Register

DAC Value	Nominal 1553 Bus Voltage (peak-to-peak)
F0 h	21.5
E0 h	20
D0 h	19
C0 h	17.5
B0 h	15.5
A0 h	14
90 h	12.5
80 h	10.5
70 h	9
60 h	7.5
50 h	6.5
40 h	4.5
30 h	2.9
20 h	1.1
10 h	0.4
00 h	0

Error Table

The Error Table is a region in RAM used by the microcode to store information on 45 possible errors. Each entry in the table, shown in Table 4-4, consists of two words and is identified by the address of the first word. The first word provides information (such as a pointer offset to a BC data structure) about the error. The second word is an error counter which indicates the number of times this error has occurred.

Table 4-4 ABI Error Table

Word Address	Byte Address	Class	Description
100h	200h	H	This error indicates an initialization failure of some of the internal logic after power up.
102h	204h	H/P	This error results either from a hardware failure or from severely distorted transmissions on the bus. It indicates that a data word was not received after detection of bus activity.
104h	208h	M	This error indicates that the microcode has entered an invalid state.
106h	20Ch	P	A word that did not have a command sync was received after a transmit command.
108h	210h	P	A word with a data sync was received when expecting a status word for a receive command.
10Ah	214h	P	This error typically is a side effect of other errors on the bus. It occurs when the microcode is waiting for a new command. Any word received that is not a command sync causes this entry to be updated. The microcode was expecting a command sync, but received a data sync.
10Ch	218h	H/P	This error results either from a hardware failure during a transmit command or from severely distorted transmissions on the bus. It occurs when bus activity is detected through a command sync but a completed transmit command is not received. The microcode receives a SARDY discrete, but never receives the required DRDY discrete.
10Eh	21Ch	P	An apparent transmit command was partially received, but contained a parity error when completed and was not processed.
110h	220h	I	A transmit command has been received for an simulated or real-time monitored RT, and no buffer has been defined in the buffer pointer table for the indicated subaddress.
112h	224h		Undefined
114h	228h	H/P	This error results either from a hardware failure during a transmit command or from severely distorted transmissions on the bus. It occurs when bus activity is detected through a command sync but a completed transmit command is not received. The microcode receives a SARDY discrete, but never receives the required DRDY discrete.
116h	22Ch	P	An apparent receive command was partially received, but contained a parity error when completed and was not processed.

Table 4-4 ABI Error Table			
Word Address	Byte Address	Class	Description
118h	230h	P	A receive command was received when only a transmit command is valid (that is, the second command in an RT-RT message sequence).
11Ah	234h		Undefined
11Ch	238h	H/M	This error indicates that the microcode has entered an invalid state
11Eh	23Ch	H/P	An apparent data word was received, but it contained a parity error
120h	**240h**	**P**	**An unexpected gap has occurred in a message. This situation may result from too few data words in a message or from non-contiguous data, This error may also indicate a "no response" condition of an RT. This is the most common error condition.**
122h	244h		Undefined
124h	248h	M	This error indicates that the microcode has entered an invalid state.
126h	24Ch	H/P	This error results either from a hardware failure or from severely distorted transmissions on the bus. It indicates that an expected receive command was not received after detection of bus activity on the second RT-RT command
128h	250h	P	A gap was detected on the bus after a receive command. The receive command should have been followed by a transmit command or by data with no gap.
12Ah	254h	I	A receive command has been received for an simulated or real-time monitored RT, and no buffer has been defined in the table of buffer pointers for the indicated subaddress.
12Ch	258h	P	An unexpected gap followed a receive mode command containing a data word.
12Eh	25Ch	P	After a receive mode command with data, a word with a command sync was received.
130h	260h	P	An apparent mode command was received, but it contained a parity error and was not processed.
132h	264h		Undefined
134h	268h	P	A mode command was received containing an illegal MIL-STD-1553B mode code.
136h	26Ch	P	A data word was expected but a command word was received.
138h	270h	I	Interrupt buffer overflow. Too many interrupts received before host servicing.
13Ah	274h		Undefined
13Ch	278h	I	A number of entries in the pointer table must contain non-zero values for proper operation. This error is incremented for each entry that is "0" when the CMD word is set to "1". The address of the last entry is placed in the parameter word associated with this error.
13Eh	27Ch	I	Bus Controller program block word error.
140h	280h	P	No gap preceded an apparent received-status word.

Table 4-4 ABI Error Table

Word Address	Byte Address	Class	Description
142h	284h	H	The transmitter was unable to send data on the bus.
144h	288h	I	HOB buffer has error (block transfer only).
146h	28Ch	P	RT-RT word count error.
148h	290h	P	Mode code timeout while waiting for data.
14Ah	294h	P	Invalid data on receive status.
14Ch	298h	P	Invalid data on last data word of BC-RT message.
14Eh	29Ch	M	Found DRDY without SARDY
150h	2A0h	P	No gap after last word of RT-BC.
152h	2A4h	P	Intermessage gap error.
154h	2A8h	P	No gap timeout detected for delay.
156h	2ACh	P	BC-RT long message error. (No gap at end of data).
158h	2BOh	H	Timer busy

Legend:
H Internal hardware failure
I Application initialization or response related
P Bus protocol
M Microcode

Hardware Reset

If the hardware reset button on the host system is pressed, the CSR is reset and microcode execution is halted. At this point, memory above "003Fh" is still intact and may be accessed. After a hardware reset, the microcode must be restarted (See "Microcode Initialization" subsection) and memory above "003Fh" will be cleared at this time.

The ABI-V2 has an optional jumper which enables the reset function. If this jumper is not set, the ABI-V2 will not be affected by a hardware reset.

NOTE: The ABI-PC is not affected by the "Ctrl-Alt-Delete" soft reset.

MICROCODE INITIALIZATION

When an ABI module is powered-up or reset, the host software must perform the proper startup procedure to initialize the microcode program. However, before performing any initialization procedure, the user must first determine which microcode program to execute. Table 4-5 lists the two major types of ABI microcode programs: Built-In Tests (program numbers 1-7) and Application Main Body (program number 13).

Table 4-5 Microcode Program Identification

Microprogram ID #		Microprogram Description	Where Discussed
1	0001 h	Built-In Test 1	Built-In Test Routines
2	0002 h	Built-In Test 2	
3	0003 h	Built-In Test 3	
4	0004 h	Built-In Test 4	
5	0005 h	Built-In Test 5	
6	0006 h	Built-In Test 6	
7	0007 h	Built-In Test 7	
8	0008 h	Reserved	--------
9	0009 h	Reserved	--------
10	000A h	Reserved	
11	000B h	Reserved	
12	000C h	Reserved	-------
13	**000D h**	**Full Function 1553B**	**"1553 Microcode Program Startup Procedure" in this manual subsection**
14	000E h	Reserved	-------
15	000F h	Reserved	

The first type of microcode program is for Built-In-Test functions. These functions are described in the "Built-In Test Routines" section of this manual. The Application Main Body program is the entry point for the "Full Function 1553B" microcode program that executes the BC, RT, Monitor and Interrupt functions of this interface.

The ABI can support two separate and independent microcode programs. This is possible due to a bit in the CSR which selects either the upper or the lower area of PROM for storage (See Table 4-2). The desired area is selected as part of the "1553 Microcode Program Startup Procedure" which follows in this subsection.

ABI-PC2 Hardware Initialization (VME Systems Skip This Procedure)

For ABI-PC2 modules, a hardware initialization step must be performed in order for the PC system to access the desired block of ABI RAM. (For information on memory configuration, see "Setting the ABI-PC2 RAM Base Address in PC Memory Space" in the "Hardware Installation" section of this manual.) This selection allows the PC system to access either 128 kilobytes or 64 kilobytes of ABI RAM at a time. To set the access mode, write the following bit values to the I/O Control Register:

1. Bit 0: Set this bit to "1" to enable the board. Upon power-up or reset, this bit is set to "0" which disables the board. This allows multiple boards to use the same memory base address.

2. Bit 1: Set this bit to "1" for 64-kilobyte access or set to "0" for 128-kilobyte access.

3. Bit 2: ABI memory has 64K X 16 RAM. If bit 1 is set to "1", bit 2 selects either the lower or the upper half of this memory. Set this bit to "0" to access the lower half of ABI RAM and set to"1" to access the upper half. If bit 1 is set to "0", this bit is ignored. The I/O Control register location is user selected (3XXh), with "300h" being the first available address.

1553 Microcode Program Startup Procedure

To start a desired ABI microcode program, perform the following procedure:

1. Write "0001" to the CSR (offset 0000h) to reset the ABI hardware.

2. Write "000D" to the MAP Register (offset 0010h) to address the "Full Function 1553B" microcode program.

3. Write "0000" to the PSTEP Register (offset 003Fh) to latch the microcode instruction for execution.

4. Write the desired operation mode to the CSR. ("0140h" is a typical value.) See Table 4-2 for CSR bit descriptions.

    - Set the **run** bit (bit 1) to "0".
    - Set the **operation enable** bit (bit 6) to "1".
    - Set the **interrupt** bit (bit 7) to "1" to clear any interrupts.
    - Set the **disable I/O** bit (bit 8) to "1".
    - Set the **prom select** bit (bit 9) to "1" if executing upper-half microcode loads; set it to "0" for lower-half microcode loads.

5. Write "0000" to control register offset 0040h. Upon proper startup of the application microcode, this register will be loaded with the microcode version number.

6. Logically "OR" the CSR value from step 4 with the value "0002" and write result to the CSR. This will set the ABI to run mode.

7. Go into a software loop, checking the control register (offset 0040h) for a non-zero value. Exit the loop when the register contains a non-zero value.

8. Load data structures and control registers into ABI memory.

9. Set the CMD control register (offset 0080h) to "0001". This enables 1553 bus processing.

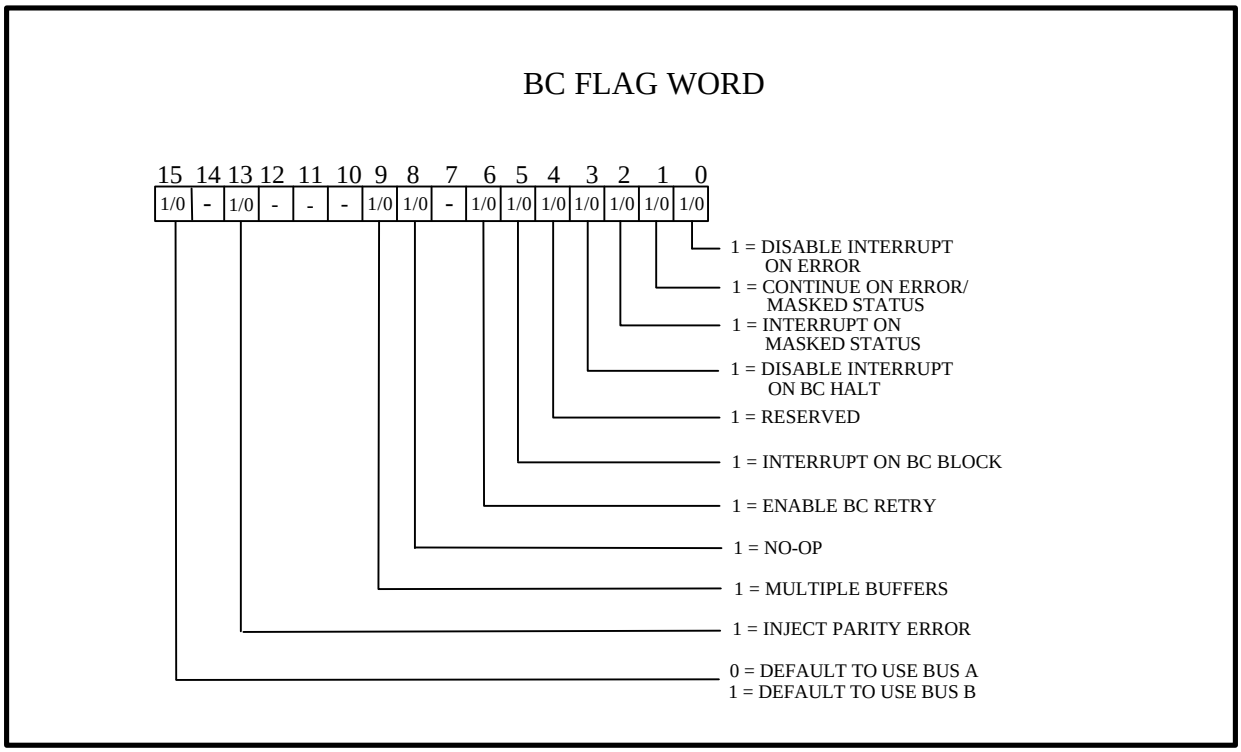
10. Verify that the RESP control register (offset 0081h) is incrementing. This ensures that the microcode has successfully performed startup and is currently processing 1553 data.

When this startup procedure is complete, control registers 0040h and 0041h will contain the microcode product and version numbers (see the cover page of this manual for these codes). Please read these registers and have their values when calling SBS Engineering with product operation questions. As step 10 above describes, the RESP register can be checked any time after startup for an incrementing number. (Bus traffic may cause RESP to increment slowly.) If this register is not incrementing, the microcode has "crashed". This is probably due to an

improper read from or write to a reserved control register or from an incorrectly programmed data structure.

BC ERRATA

This Errata details new features for the Bus Controller Simulation mode of the ABI-PC2. Three bits (5, 8, and 13) have been added to the BC Flag Word to support these features.



BC Flag Word

New Flag Word Bits

Bit 5 - Interrupt on BC Block
Set this bit to "1" to generate interrupt code 12 upon execution of the BC block.

Bit 8 - No-Op
Set this bit to "1" to cause the microcode to ignore all actions for the BC block. When the microcode encounters a BC block with the No-Op bit set to "1", it immediately attempts to process the next block. If the next block has the No-Op bit set, the microcode proceeds to the following block until it reaches a BC Control Block which does not have the No-Op bit set.

Bit 13 - Inject Parity Error
Set this bit to "1" to inject a parity error on the last word of the BC transmission. The error will be injected on the last data word, if data words are transmitted; otherwise, the error will be applied to the command word.

BUS CONTROLLER SIMULATION

The ABI is capable of full Bus Controller (BC) simulation while performing RT simulation and Sequential and Map Monitoring. BC simulation in the ABI is governed by a single data structure consisting of a linked list of command blocks. Each command block in the linked list contains encoded information for the transmission of different BC command messages. This linked-list can be loaded at any location in the data structure memory area (0400h to FFFFh). Each command block of the linked-list consists of eight words, containing the following information:

- A Type word designating the type of command block: either BC message transmission type or time delay block type.
- Two command words to be transmitted.
- Registers to store the status-response words transmitted by remote terminals at the completion of each operation.
- A pointer to a data buffer which stores the data from the bus transaction (if the command block is a time delay function, this word will contain a "time ticks" value.)
- A pointer to the next block in the chain.

The host computer constructs a linked-list data structure of command blocks and loads this structure into ABI memory. It then programs the register BCIPTR (offset 008Fh) with the address of the first word of the first command block. During the idle routine of the microcode 1553 program, the register BCIPTR is monitored. When the microcode finds a non-zero value, it uses the value as a pointer to the first command block of the linked-list. It sets BCIPTR to "0" and begins to transmit the BC command message as it is encoded in each command block. (There may be up to a 20 microsecond delay from the time the host software writes the value to BCIPTR to when the first BC transmission begins.) The host software can monitor BCIPTR to determine when BC transmission begins.

The BC chain data structure is one of the few data structures that does not need to be defined in ABI memory at microcode startup.

Note: The BC linked-list data structure is also referred to as the **BC chain** or **chain program**.

Control Block Structure

ABI transmitted BC command messages are specified by the host system in a series of linked-list command blocks. The host software can construct any length command block linked-list that ABI memory will allow (over 1500 blocks if the ABI is performing only BC simulation). Figure 4-4 describes the data structure for BC command block linked-list chains.

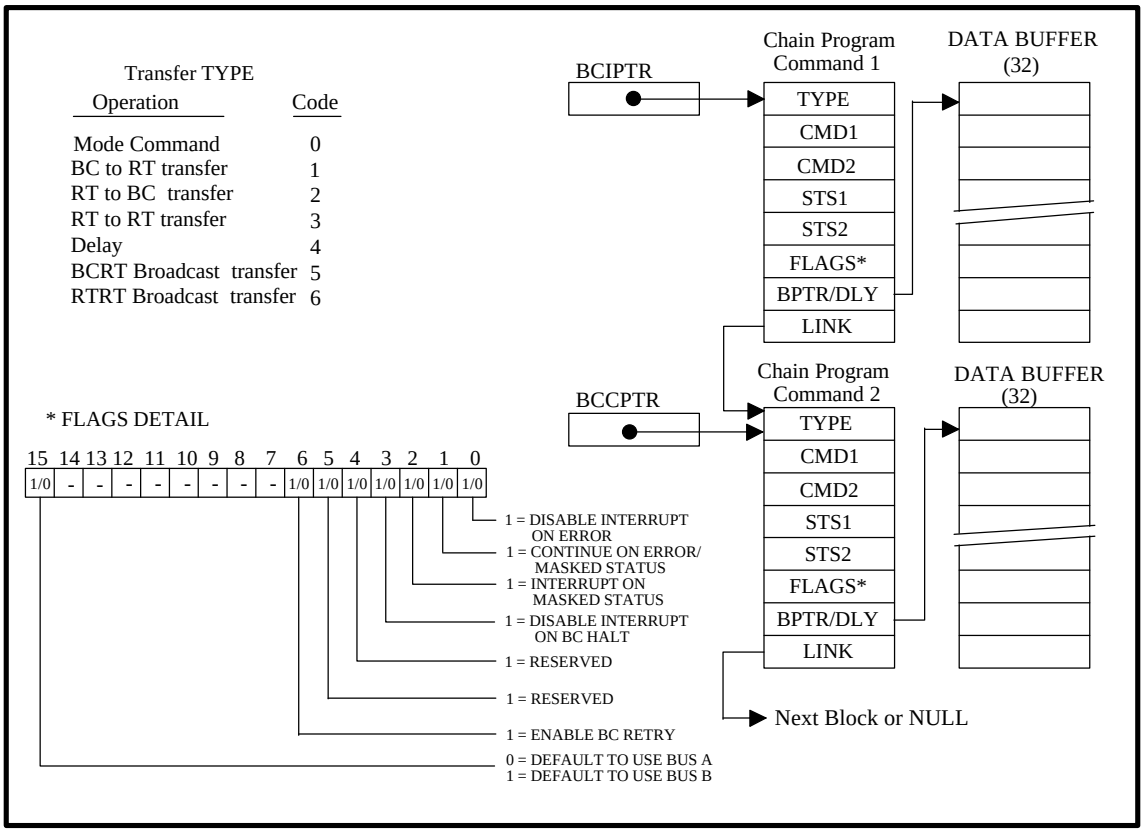


Figure 4-4 BC Control Structures

The definition of each word is presented below.

Type
A code describing the type of bus-control operation to be performed (BC-RT, RT-BC, RT-RT, Broadcast, etc). The available codes are shown in Table 4-6.

Table 4-6 BC Type Codes

Operation	Code
Mode Command	0
BC to RT Transfer	1
RT To BC Transfer	2
RT to RT Transfer	3
Delay	4
BC to RT Broadcast Command	5
RT to RT Broadcast Command	6

CMD 1
The command word associated with a particular operation, or the first command word for an RT-RT transfer.

CMD 2
The second command word for an RT-RT transfer.

STS 1
The status word returned by the remote terminal in response to an operation, or the second status word returned in an RT-RT transfer from the receive command word (CMD1).

STS 2
The first status word returned in an RT-RT transfer. This status word is associated with the transmit command word (CMD2).

FLAGS
A set of flags indicating the response the ABI system takes to certain bus control error conditions. The bus controller also controls which bus (A or B) is used for transmitting commands. The first three bits in the flag word for a chain program block determine how the ABI microcode treats errors and status word returns encountered during bus controller operation. The first two bits define how errors are handled.

Bit 0- Disable Interrupt on Error
Setting this bit to a "1" disables generating interrupts on any error conditions. Setting this bit to "0", the default condition, results in the microcode generating an interrupt for any error encountered. *Note: The proper error code in the Error Table is updated regardless of the value of this bit.*

Bit 1 - Continue on Error/Masked Status
Determines whether the microcode ceases execution of the chain program upon encountering an error or masked status. Setting this bit to a "1" causes the microcode to continue executing the chain program after encountering an error or interrupting on masked status. Setting this bit to "0" results in the microcode terminating the chain program upon encountering any errors or interrupting on masked status.

Bit 2 - Interrupt on Masked Status
Determines which status word conditions cause the bus controller portion of the microcode to interrupt the host system. If this bit is set to "1", status responses to the bus controller are compared to the bus controller status word mask (BCSMSK) entry in the pointer table. If any bits set to "1" in the returned status word/words match bits set to "1" in the bus controller status word mask, the microcode interrupts the host processor. When this occurs, the microcode reads the Continue on Error/Masked Status flag to determine whether to halt the chain program.

Bit 3 - Disable Interrupt on BC Halt
Disables interrupt 06h when the BC chain execution halts. Halting of a BC chain most commonly occurs due to a zero link in the last control block of a chain, but may also occur due to an error or an interrupt on masked status.

Bit 6 - Enable BC Retry
Enables BC retry for this block.

Bit 15 - Bus Select
Tells the microcode to send the command on bus A or B. If this bit is a "0", the default setting, the microcode sends commands on bus A. If the bit is set, the microcode uses bus B.

BUFPTR
Points to a data buffer which is to provide transmitted data or accept received data. For delay blocks, this word contains the delay interval in clock ticks. Determine the clock tick as follows:

1. Determine the MIPS rating of the ABI:

   MIPS=10 for ABI-V2, ABI-PC, and ABI-Q
   MIPS=15 for ABI-V3 (serial numbers greater than 200) and ABI-PC2

2. Divide this rating into "1".

For example, if the MIPS value is 15, the clock tick value is 67 nanoseconds (or 1/15,000,000).

> Note: It takes 15 microseconds to process a delay block. Add this to the delay interval value (clock tick value) when determining the intermessage gap time.

LINK
Link pointer to the next command block in the chain program. A zero value indicates the end of the linked-list.

During bus controller operation, the microcode decodes each block of the linked-list program and takes the appropriate action. If a BC-RT, RT-BC or mode type is indicated, CMD1 is transmitted on the bus. Any resulting status is placed in STS1. BUFPTR defines the buffer from which data is transmitted, or into which data is received. If an RT-RT transfer is specified, then CMD1, CMD2, STS1, and STS2 are used. BUFPTR is not used when RT-RT is specified.

Controlling BC Operation - BCIPTR, BCCPTR, BCLPTR

To operate the ABI in bus controller mode, set up the appropriate chain and data buffers in ABI RAM. Initiate chain program execution by loading BCIPTR with the offset of the first command block. As soon as the ABI begins to execute the chain program, it clears the contents of BCIPTR. While the chain program is executing, the ABI updates the BCCPTR entry in the Control Register table (offset 0090h) so that it points to the chain program block currently being executed. Execution of the chain program continues until a "0" is encountered in the LINK word of a chain program command block. If the LINK in the last command block of the chain program points back to the first command block of the linked-list, the ABI operates in a continuous-loop mode. Halt or redirect chain program execution during operation by entering a new value in the LINK word of a particular command block, prior to execution of that block. Enter a "0" to halt execution or a different offset to change the execution path. This would allow special BC messages to be inserted into the minor frame for cases such as the service requested bit being set by the RT Status Word. The control register BCLPTR (offset 0091h) points to the last BC block that was executed.

Status Word Mask - BCSMSK

The control register BCSMSK (offset 0092h) is used as a status word filter to allow the BC to interrupt when a certain status response is received. This feature is based on two values: 1) bit 2 in the FLAG word of the BC control block and 2) the entry of BCSMSK. When bit 2 of the FLAG word is set to "1", the Returned Status word is logically "ANDed" with the BCSMSK word. If the resulting value contains any bits set to "1", an interrupt is generated. The microcode then reads the Continue On Error/Masked Status flag to determine whether to halt the chain program.

BC Retries - BRTCNT, BRTBUS, BRTCMD, BRTRTC

A BC program block can be set up to retry the message if there is an error in the RT response. The retry feature is enabled by setting bit 6 of the FLAGS word (in the BC block shown in Figure 4-4) to "1". There are four control registers for BC retries: BRTCNT (offset 00AFh), BRTBUS (offset 00B0h), BRTCMD (offset 00B1h), and BRTRTC (offset 00B2h). BRTCNT specifies the maximum number of retries (a maximum of 16 is allowed).

BRTBUS is a word which defines which bus (A or B) will be used for the retried message. Each bit of this word pertains to a particular retried message: the first or least-significant bit applies to the first retry of the message, the second bit applies to the second retry of the message, and so on, up to the bit which applies to the number of retry messages defined by BRTCNT. If a bit is set to "0", the associated retry of the message is retried on the same bus as the original message from which the error was detected. If a bit is set to "1", the associated retry of the message is sent on the alternate bus. For example, if the first message was sent on Bus A and the retry bit is a "1", the retry of the message will be sent on Bus B.

BRTCMD contains the command that was last retried. The value is set to "0" at the start of each new BC chain.

BRTRTC contains the actual number of retries. It is set to "0" at the start of each new BC chain.

NOTE: Retries are always attempted before the state of the Continue on Error/Masked Status bit (bit 1) in the BC FLAG word is checked. Therefore, if bit 1 is set to "0", retries will be attempted before execution of the BC chain is halted; if bit 1 is "1", retries will be attempted and normal BC chain execution will continue.

Mode Codes

The ABI processes mode codes in the same way a normal bus controller handles them. For mode code commands without data words, the microcode simply transmits the command word. For mode code commands with data words, the data word is transmitted or stored as the first word of the command block data word buffer.

REMOTE TERMINAL SIMULATION

This subsection of the manual reviews the key data structures required for ABI RT simulation. The ABI is capable of simulating all 31 possible remote terminals (RT) while performing BC and Monitoring functions.

The ABI provides great flexibility for RT simulation. Linked-lists of data word buffers are used for both the transmission of RT messages and receiving of BC transmitted messages. The linked-list buffers may be chained indefinitely, limited only by the available ABI memory. (If the ABI is performing only RT simulation, nearly the entire 64K word memory range is available.) The ABI also provides interrupt capabilities for each of the 31 RT addresses and their related 64 subaddresses (32 transmit and 32 receive buffers). This provides the host system with real-time access to RT data buffers for the most complex simulation applications.

ABI Control Registers BCIGP, RSPGPA and RSPGPS may be used to change the **intermessage and status response gap times** and are described at the end of this subsection.

Note: There are areas in this subsection which overlap the "Bus Monitoring" subsection. RT Map Monitoring utilizes the same linked-list buffers as RT Simulation for real-time storage of data words. This subsection will discuss these buffers as they relate to RT Map Monitoring. Please also read the "Bus Monitoring" subsection for a complete understanding of Map Monitoring functions.

The key data structures for RT Simulation are:

- **The Status Word Table** which enables desired RTs for simulation and stores status response information.

- **The RT Address Data Structure** which enables the status responses for the RTs enabled in the Status Word Table and provides the leading address for the linked-list buffers (for both RT simulation and RT Map Monitoring).

- **The Filter Table** which provides interrupt enabling, parity error generation, and Sequential Monitoring control bits (see "Bus Monitoring" subsection).

Other important data structures include:

- The **Protocol Table** which selects different RT responses, such as "1553A subaddress 31", Dynamic Bus Control Accept, etc.

- The **Mode Code Tables** which contain dedicated data structures for data word mode codes.

This subsection details each of these data structures.

Status Word Table - Enabling RT Simulation

The Status Word Table is a 64-word table containing two entries for each of the possible 32 remote terminals being simulated. It provides the ABI microcode with information about the remote terminals currently being simulated, and stores information used during a Remote Terminal Status Response. The offset of the Status Word Table is loaded by the host software into the SWTPTR control register (offset 0088h). Figure 4-5 shows the basic structure of this table and the bit definitions of a status word.

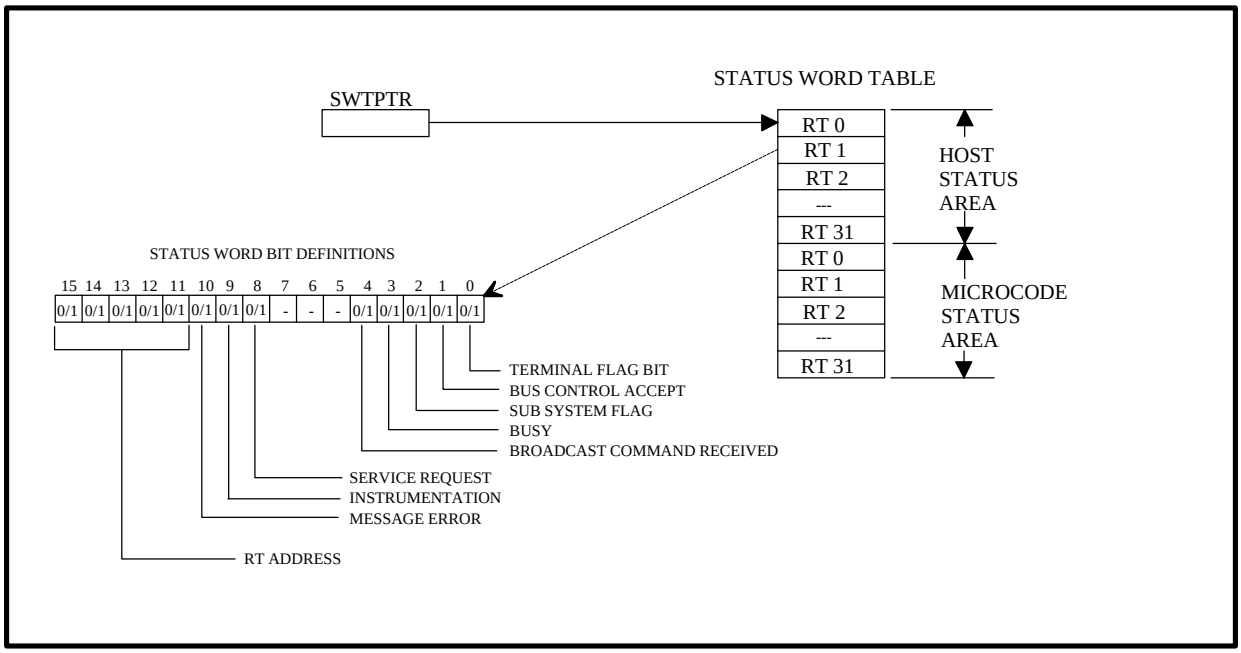


Figure 4-5 RT Status Word Table (SWTPTR)

One RT entry is contained in the **host** status area and the other in the **microcode** status area. The status word in the host status area is initialized and maintained by the host software. **The status word in the microcode status area is maintained by ABI microcode and should not be modified by host software.**

The **host** status word performs two functions: 1) It indicates to the ABI that a particular RT is active if the associated host status word contains a non-zero entry and 2) It is logically "ORed" with the microcode status word to obtain the transmitted status response word. Any bit set to "1" in the host status word will be set to "1" in the transmitted **status response** word (with the exception of the **message error** bit, described below).

Normally, the host status word only contains the address for a given RT. A non-zero address value must be loaded into the table to enable a particular RT. *To enable RT 0, the host sets the message error bit (in the host status word) to "1".* This action does not affect the status response word because the message error bit actually transmitted is taken from the microcode status area.

The other bits in the status response word are assigned a value by a logical "OR" operation involving the host status word and the microcode status word. Therefore, the host may force any bit in the status response word to "1" by setting it to "1" in the host status word (with the exception of the message error bit). An erroneous RT address could be entered, for example. The function of each bit in the status response word and a description of how the bits are generated is presented below.

Note: Those bits which are normally set by the hardware during 1553 operation are handled by the ABI microcode. Those bits normally set by sub-system software during 1553 operation are handled by the host.

Status Word Response Bits

Bit 0 - Terminal Flag Bit
The host software sets this bit to "1" to indicate a "terminal-failed" condition. The ABI microcode also controls this bit when responding to Inhibit Terminal Flag Bit and Override Inhibit Terminal Flag Bit mode commands. The microcode responds to an Inhibit Terminal Flag Bit mode command by sending a status word containing a terminal flag bit set to "0", regardless of the actual status of this bit. The microcode responds to a subsequent Override Inhibit Terminal Flag Bit by sending a status word containing a terminal flag bit with the actual state of this bit in the microcode status word table. Please see "RT Phase Data Structure", under "RT Mode Code Simulation" in this subsection, for more details.

Bit 1 - Bus Control Accept Bit
This bit is set by the ABI microcode in response to a Dynamic Bus Control mode command. (The state of the bit is determined by the state of bit 2 in the Protocol Table entry for the particular RT. If the bit is set to "1" in the Protocol Table, the **bus control accept** bit will be set to "1" in the status response word.) The host software must also respond to this mode command by enabling an interrupt so that it may initiate a bus controller program. Both of these actions must take place in order for Dynamic Bus Control to be accepted.

Bit 2 - Subsystem Flag Bit
This bit is set to "1" by the host software to indicate a potentially failed subsystem. It is never set by the ABI microcode.

Bit 3 - Busy
This bit is set to "1" by the host software to indicate a "terminal-busy" condition. It is never set by the ABI microcode. When this bit is set to "1", the RT will not transmit any data words, only status words.

If the busy-bit is set in the status word of a receiving RT involved in a BC-RT or RT-RT message, data will be stored in the RT data buffer. Upon successful

completion of the BC-RT or RT-RT message, the RT will transmit a status word with the busy bit set.

When the busy-bit is set in the status word of a transmitting RT, the resulting action depends on the message type. For both an RT-BC and RT-RT message, the RT will transmit its status response with the busy-bit set and will suppress the transmission of data words. For an RT-RT message, the receiving RT will detect that no data were transmitted and wait for a retry by the transmitting RT. The receiving RT will not transmit a status word while waiting for the retry. For RT's involved in mode codes, the transmission of an associated data word will be suppressed.

Bit 4 - Broadcast Command Received Bit
On receipt of a broadcast command, the ABI microcode sets this bit to "1" for all active RT simulations. In response to a Transmit Status Word and/or Transmit Last Command mode code immediately following a broadcast command, this bit will be set in the RT's status response word. Otherwise, the microcode will reset this bit to "0".

Bit 8 - Service Request Bit
This bit in the host status word is set to "1" by the host sofrware; thereafter, this bit appears in each status response word until it is cleared by the host software. The host software should respond by enabling an interrupt on a Transmit Vector Word mode command and clearing this bit (in the host status word) as part of the interrupt service routine.

Bit 9 - Instrumentation Bit
This bit is used by the host software for specialized applications. It is never set by the ABI microcode.

Bit 10 - Message Error Bit
This bit indicates that the previous message received by the RT did not pass all validity checks. It remains set to "1" in the status word table until the next message is received, at which time it is updated to reflect the validity of that next message. It is set only by the ABI microcode. The proper code in the Error Table is updated when an invalid message is detected.

RT Address Data Structures

Defining Data Word Buffers

The RT Address Data Structures consist of the Address Table, Buffer Pointer Table and linked Data Buffers. The control register ATPTR (offset 0089h) contains the address of the first word of the Address Table. The Address Table provides addresses to the Buffer Pointer Tables associated with each RT. The Buffer Pointer Table gives the location of the first linked-list buffer for each RT subaddress (receive and transmit). The size of these linked-lists

are limited only by the available ABI memory. Figure 4-6 illustrates the RT Address Data Structures.

These RT Address Data Structures are used for both RT Simulation and real-time Map Monitoring. The Address Data Structures are programmed identically for both simulation and monitoring functions, and the ABI can simulate and Map Monitor different RT messages within these data structures simultaneously.

The same RT message cannot be both simulated and Map Monitored. (This would not be a useful function because the Data Words would just be duplicated.) Sequential Monitoring provides for self-monitoring of simulated and external messages. The following paragraphs explain programming the ABI for RT message simulation. However, the information applies to setup for RT Map Monitoring, including Broadcast Commands with Receive Data Words.

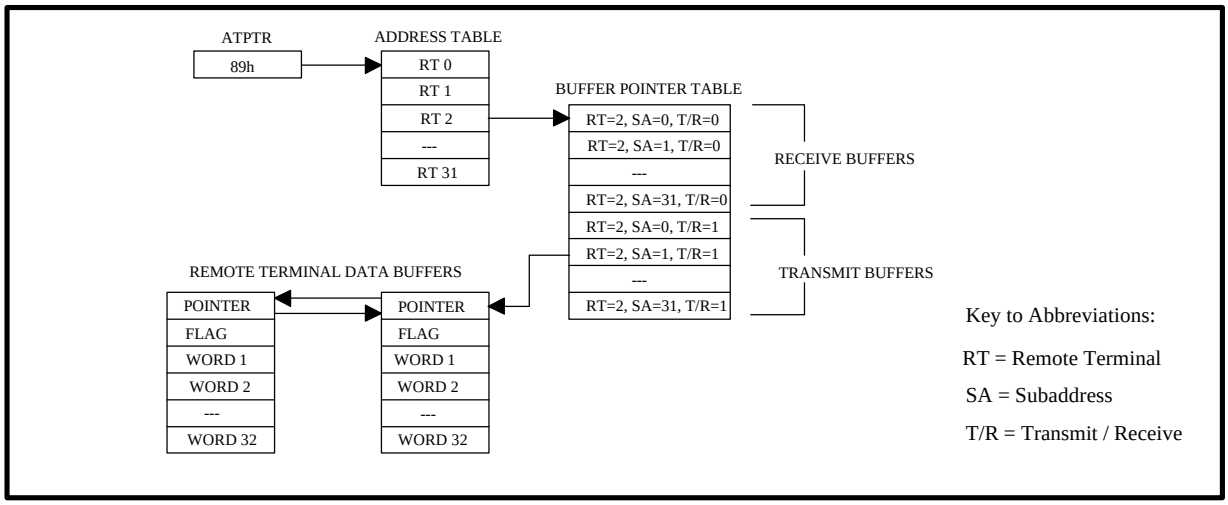


Figure 4-6 RT Address Data Structures

Address Table

The Address Table is 32 words long and holds the Buffer Pointer Table address for each of the 32 Remote Terminals. The Buffer Pointer Table address of an RT must be programmed into the address table prior to RT Simulation or Map Monitoring. An RT not being simulated or monitored must have a zero value programmed into the respective element of the address table. When the ABI microcode detects a 1553 command word on the bus, it checks the buffer pointer table location (respective to the command word Terminal Address) programmed into the Address Table for a non-zero value. If the value is "0", the ABI will not respond with status or data words. If there are no data buffers defined and the appropriate element of this structure is "0", the ABI will not respond with a status word.

For example, if RT 6 is to be simulated, element 7 of the Address Table must contain the address to the Buffer Pointer Table for RT 6. If RT 6 is the only RT to be simulated, all other elements must have a value of "0".

If there are no data buffers defined and the appropriate element of this structure is "0", the ABI will not respond with a status word (in Map Monitor mode, the microcode will simply ignore the message and not store the command or data words).

Buffer Pointer Table

The Buffer Pointer Table has one entry for each possible receive subaddress and one for each possible transmit subaddress. The first 32 pointers correspond to receive (T/R bit = 0) subaddresses 0-31, and the second 32 pointers correspond to transmit (T/R bit = 1) subaddresses 0-31. Each of the 64 elements may point to the beginning of a linked-list of data buffers which is used to transmit or receive the RT bus data words. A "null" entry in this table indicates that the corresponding subaddress is not active for this particular RT (no data buffers are defined).

Data Buffer Format

Each data buffer consists of 34 words of memory. The last 32 of these 34 words are used to store RT message data words. The first two words provide special features. The first word is a **pointer** that may provide the address to another data buffer. The second word in the buffer is called the **flag** word.

The link of buffers joined together by the **pointer** words may be as large as ABI memory allows. Usually the last data buffer points back to the first data buffer to create a circular chain. Figure 4-6 shows two buffers that alternate in this manner. If only one buffer is required, which is often the case, the first word must contain its own address (points back to itself). After transmission of data words to a buffer, or upon receiving data words that are stored in a buffer, the ABI microcode copies the Data Buffer Pointer address into the respective Buffer Pointer Table location. The ABI microcode always uses the address stored in the Buffer Pointer Table as the target address for storage or transmission. This allows the host computer to monitor the Buffer Pointer Table and determine the current location of the microcode in the data buffer linked-list.

After a valid message transfer is complete, the **flag** word is loaded with the command word portion of the message. If the message is not valid, this word will be set to "0" (in most cases the data words will be written to the buffer upon error, but the microcode cannot guarantee proper values or word count). The host program should set the flag word to "0" after reading the buffer. This permits the programmer to determine whether new data has been received at this subaddress by looking for a non-zero value in the flag word. For variable-length messages (less than 32 words), the word count field of the command word (flag word) will tell the host system the number of valid data words in the buffer.

Broadcast Considerations

To store the data from a broadcast command, define a subaddress data buffer for RT 31 and enable the map monitor bit (bit 1) for RT 31 in the Protocol Table. Data buffers defined for

RT 31 only function as map monitor data buffers for broadcast commands (see the "Bus Monitoring" subsection of the manual for more information on map monitoring). Data from a broadcast command is never placed in the corresponding subaddress data buffers of RTs 0 - 30.

For example, assume that RT2/r/SA4 and RT4/r/SA4 are being simulated in ABI memory and each RT/SA has a receive data buffer defined for subaddress 4. Also, assume that a data buffer has been allocated for RT31/r/SA4 and map monitoring has been enabled for RT 31. If a BC-RT broadcast command is sent on the bus, the data from that bus transfer will be found in RT31/r/SA4's data buffer. The data will not be present in the subaddress data buffers of RT 2 or RT 4.

Minimum Programming Requirements for the RT Address Data Structures

The ATPTR must be programmed to point to an Address Table of 32 elements. The Address Table does not have to point to any Buffer Pointer Tables (if the ABI is not simulating or Map Monitoring Remote Terminals, all the Address Table elements will have NULL values).

Filter Table

The Filter Table Data Structures provide setup flags for Message Interrupt, Sequential Monitoring, and parity error generation for simulated RT messages. Figure 4-7 illustrates the Filter Table Data Structures.

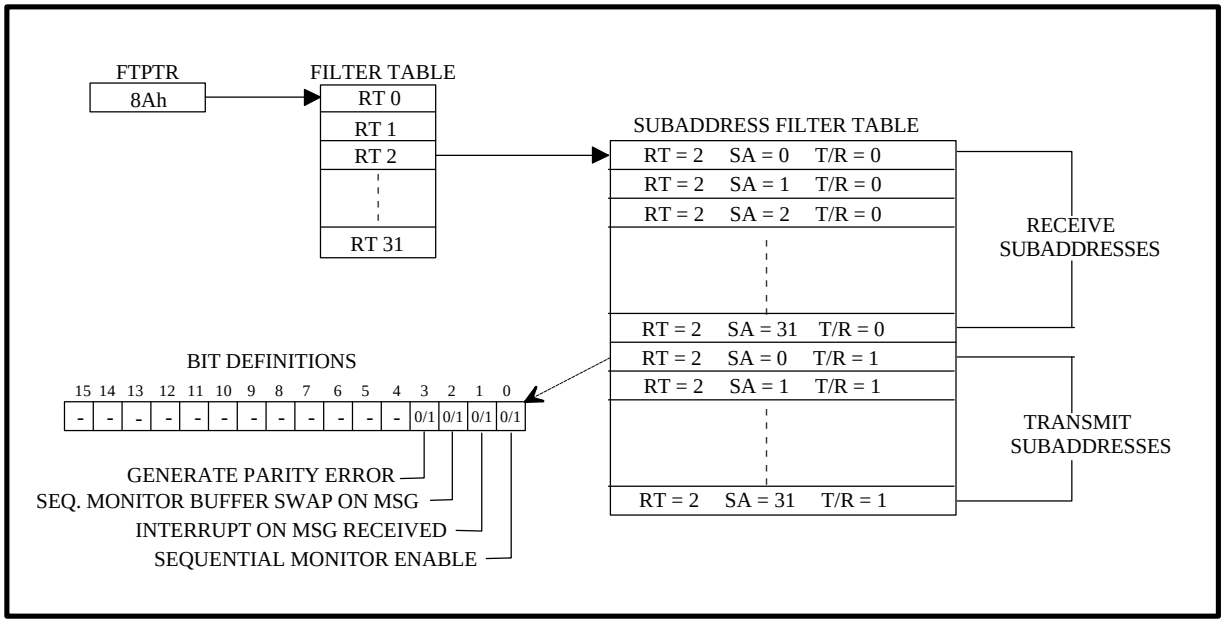


Figure 4-7 Filter Table Data Structures

The control register FTPTR (offset 8Ah) holds the address of the first word of the Filter Table array. The Filter Table provides 32 pointer elements, one for each RT address. Each element

points to an array of 64 flag words called the Subaddress Filter Table which provides operation flag bits for all receive and transmit subaddresses. This provides operational control to the subaddress level of each RT.

The following bits may be programmed for each receive and transmit flag word in the Subaddress Filter Table. Bits 4-15 should be set to "0".

Bit 0 - Sequential Monitor Enable
If this bit is set to "1", data to or from this RT and subaddress combination is placed in the Sequential Monitor Buffer. This data includes command, status, data and time stamp words.

Bit 1 - Interrupt on Message Received
If this bit is set to "1", an interrupt occurs when a command word from a simulated or external Bus Controller is detected for this RT/subaddress combination. The interrupt will occur after the message is completely received.

Bit 2 - Sequential Monitor Buffer Swap on Message
If this bit is set to "1", message activity to or from this RT/subaddress combination will cause the ABI microcode to first swap the Sequential Monitor Buffers and then generate a hardware interrupt (code 0010h). If message activity to or from this RT/subaddress combination is also being sequentially monitored (i.e., bit 0 = 1), the following actions will take place: 1) the message for this RT/subaddress combination will be placed in the active buffer as the last message, and 2) the monitor buffer pointers will then be swapped. Following the swap, hardware interrupt code 0010h will be generated.

Bit 3 - Generate Parity Error
If this bit is set to "1", this simulated RT/subaddress combination will respond with a parity error. If this RT responds with only a status word, the status word will contain a parity error. If this RT responds with a status word and data word(s), the last word will contain a parity error.

Minimum Programming Requirements for the Filter Table

The FTPTR must be programmed to point to the first word of the Filter Table. Each pointer element of the Filter Table must point to the first word of the 64 word Subaddress Table array. None of bits of the Subaddress Filter Table have to be set to "1"; if no flag bit operations are desired for the respective RT subaddress, set all bits to "0".

Protocol Table

The Protocol Table allows the user to program ABI 1553 protocol characteristics and to select which Remote Terminals will be monitored in the RT Address Data Structures for Map Monitoring. Figure 4-8 illustrates the Protocol Table.

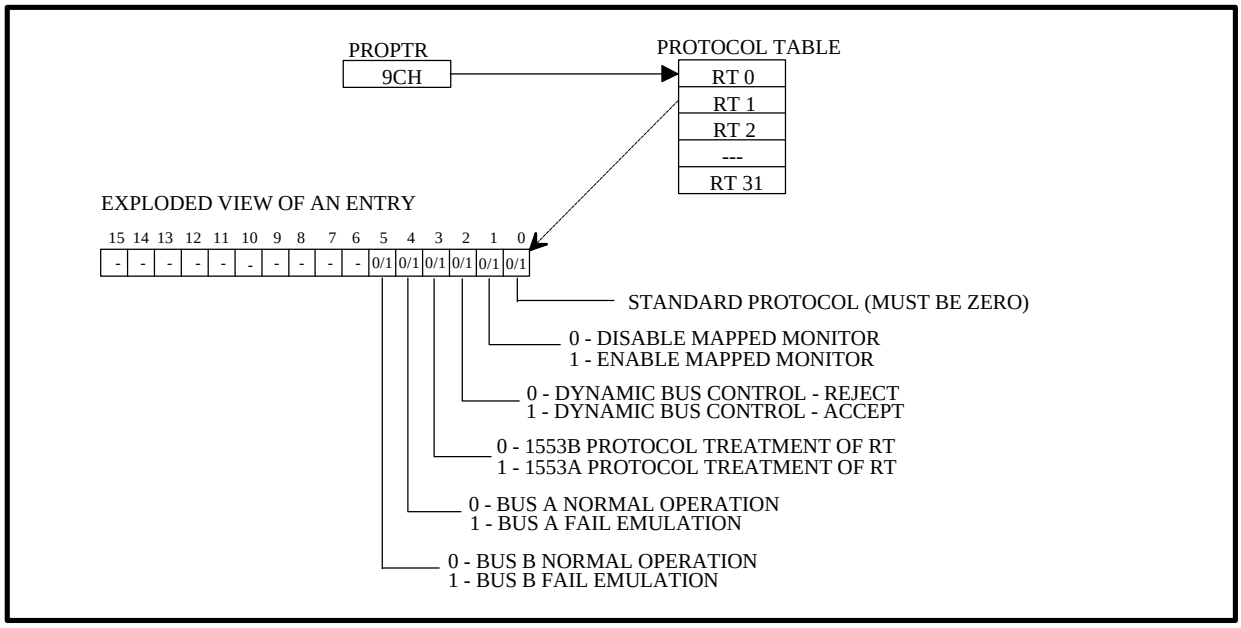


Figure 4-8 Protocol Table

The Control Register PROPTR (offset 009Ch) must be programmed with the address of the first word of the 32-word Protocol Table. Each element corresponds to a RT address and functions as a flag word. Bits can be set to control ABI protocol changes or enable Map Monitoring at the RT level. The following paragraphs describe flag bit options.

Bit 0 - Standard Protocol
Must always be set to "0".

Bit 1 - Mapped Monitor Enable
Set this bit to "1" to enable Map Monitoring for the respective RT. This causes the ABI to store received RT messages in the appropriate RT data buffer of the RT Address Data Structures. Set this bit to "0" to disable Map Monitoring for the respective RT. If the host program attempts to simulate and Map Monitor the same RT (the Status Word Table is programmed for the same RT), the Simulation function will override the Map Monitoring function.

Bit 2 - Dynamic Bus Control
This bit affects the status response to a Dynamic Bus Control Acceptance mode command. This bit is copied to the associated Status Word bit (bit 1) before the Status Word is transmitted for simulated RT responses. Please see the Mode Code discussion later in this subsection for other Mode Code program options.

Bit 3 - Treatment of RT
If this bit is set to "0", the ABI microcode handles the RT as per the 1553B specification. If this bit is set to "1", the microcode handles the RT as per the 1553A specification. Note the following items when selecting 1553A.

1553A Protocol:

RT Status Response Time: As per section 4.3.1 of the MIL-STD-1553A specification, RT status responses will be transmitted within 2.0 to 5.0 microseconds dead bus time.

Broadcast Considerations: Broadcast commands are not supported in the 1553A specification.

Mode Codes: The only defined mode code in 1553A is mode code 00000h, designated as "dynamic bus allocation". All other mode codes are user defined and are responded to with only a status response word.

Subaddress 31: Unlike 1553B, subaddress 31 is treated as a normal subaddress and standard message transfer types may designate subaddress 31 in the subaddress field of the command word.

Bit 4 - Bus A Operation
If this bit is set to "0", the simulated RT will respond properly to messages received on bus A. If this bit is set to "1", the RT will not respond to messages received on bus A.

Bit 5 - Bus B Operation
If this bit is set to "0", the simulated RT will respond properly to messages received on bus B. If this bit is set to "1", the RT will not respond to messages received on bus B.

Minimum Programming Requirements for the Protocol Table

The PROPTR pointer must be programmed with the address of the first element of the 32-word Protocol Table array. The Protocol Table flag words are programmed as required by the 1553 system. Set all flag words to "0000h" for normal 1553B operations.

RT Mode Code Simulation

The ABI responds properly to all Mode Code commands in the RT Simulation mode. For some Mode Codes, a proper response is the simple transmission of a Status Word. Other Mode Codes require actions such as the transmission or reception of data words, the setting of status word bits, etc. For many Mode Code operations, the ABI provides special data

structures to store data words or other operation words. Table 4-7 lists 1553B Mode Codes, explains the function of each, and indicates which have associated data words.

Table 4-7 ABI Mode Code Operations

Mode Code	Transmit / Receive Bit	Associated Data Word	Broadcast Command Allowed	Function
00000	1	No	No	Dynamic Bus Control
00001	1	No	Yes	Synchronize
00010	*1	No	No	Transmit Status Word
00011	1	No	Yes	Initiate Self-Test
00100	1	No	Yes	Transmitter Shutdown
00101	*1	No	Yes	Override Transmitter Shutdown
00110	1	No	Yes	Inhibit Terminal Flag Bit
00111	1	No	Yes	Override Inhibit Terminal Flag Bit
01000	*1	No	Yes	Reset RT
10000	1	Yes	No	Transmit Vector Word
10001	0	Yes	Yes	Synchronize
10010	1	Yes	No	Transmit Last Command
10011	1	Yes	No	Transmit Bit Word
10100	0	Yes	Yes	Selected Transmitter Shutdown
10101	0	Yes	Yes	Override Selected Transmitter Shutdown
* Minimum Required Mode Codes				

Mode Code Data Structures Required for ABI Operations

Last Status Word Data Structure

The status word the ABI microcode sends in response to a valid command is stored in a 32-word Last Status Word table. These status words are stored in a location corresponding to the particular RT involved in the transmission. Each time a response to a valid command is made, this table is updated to reflect the most current response. The LSWPTR control register (offset 0099h) holds the starting address of this table, as shown in Figure 4-9. The host software must allocate memory space for the table but the buffer contents must only be updated by the ABI microcode. **This is a required data structure for ABI operations.**

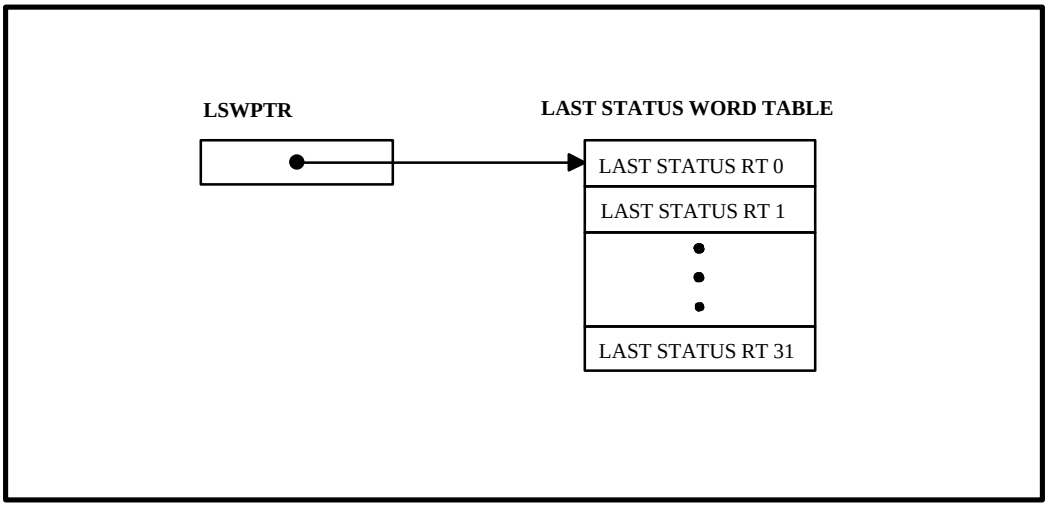


Figure 4-9 Last Status Word Data Structure

RT Phase Data Structure

Includes:

- Transmitter Shutdown
- Override Transmitter Shutdown
- Inhibit Terminal Flag
- Override Inhibit Terminal Flag

The RT Phase Table stores status information for Transmitter Shutdown and Terminal Flag Inhibit. The table is 32 words long, with one word for each RT. The RTPPTR control register (offset 0095h) holds the starting address of this table. Figure 4-10 illustrates the structure of the table.

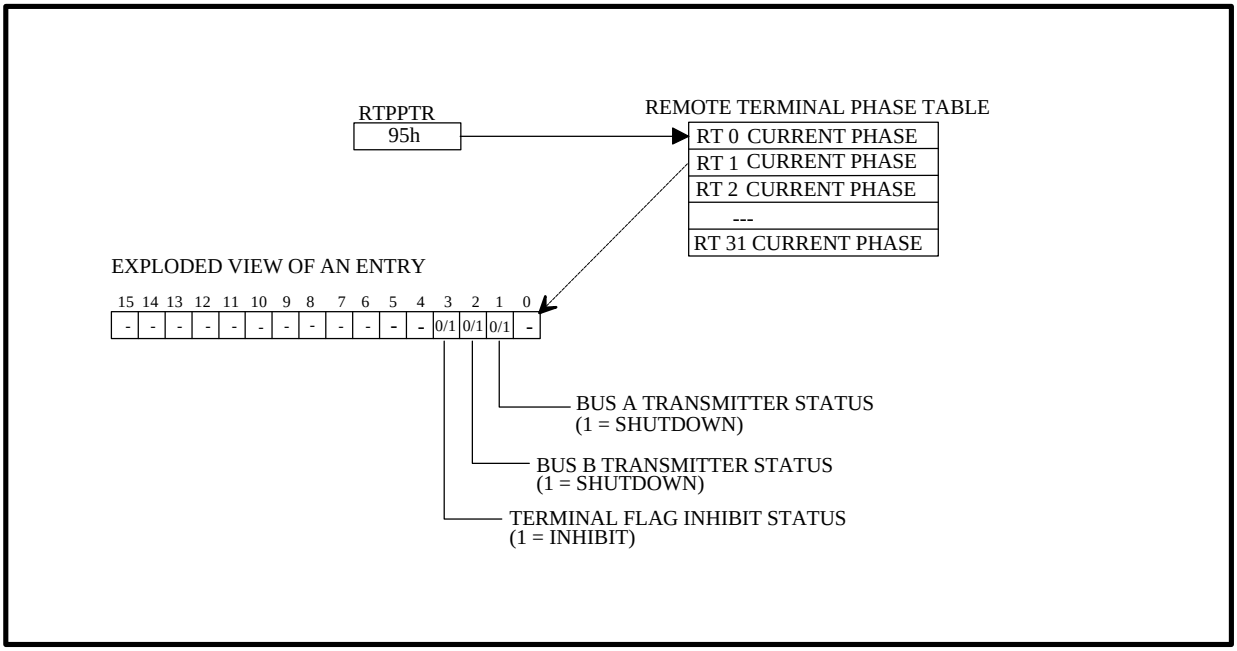


Figure 4-10 RT Phase Table Data Structure

Bits 1 and 2 of each word in the table store the shutdown status of the corresponding RT transmitter. A "1" in either bit indicates that the ABI has responded to a Transmitter Shutdown Mode Code and has shutdown the transmitter. (If the "1" is contained in bit 1, the transmitter for Bus A is shutdown; if the "1" is in bit 2, the transmitter for Bus B is shutdown) The Override Transmitter Shutdown Mode Code sets the appropriate bit to "0".

Bit 3 indicates the status of the Terminal Flag Inhibit function for each RT Status Word. When this Mode Code is received, the ABI microcode sets the appropriate bit in the table to "1" to inhibit the transmission of a Terminal Flag bit set to "1" (from the Status Word Table for the respective RT). The ABI microcode checks this bit before transmitting any status word. If it is set to "1", the microcode sets the Terminal Flag bit to "0", regardless of the actual value of this bit in the host Status Word. The receipt of an Override Inhibit Terminal Flag Mode Code

sets bit 3 to "0", allowing the Terminal Flag bit in the Status Word Table for the respective RT to be programmed. **This is a required data structure for ABI operations.**

Transmit Vector Word Data Structure

The ABI microcode responds to a Transmit Vector Word Mode Code command for a simulated RT by returning the appropriate vector word (as described in the Service Request bit discussion in the Status Word Table description). The returned Transmit Vector data word is taken from the specific RT location of this 32-word table. The TVWPTR control register (offset 0098h) holds the starting address for the table. The table array provides storage for 32 Transmit Vector words, one for each RT. Figure 4-11 illustrates the table structure. The host software always has access to this structure in order to store the desired Vector Word value for RT responses. **This is a required data structure for ABI operations.**

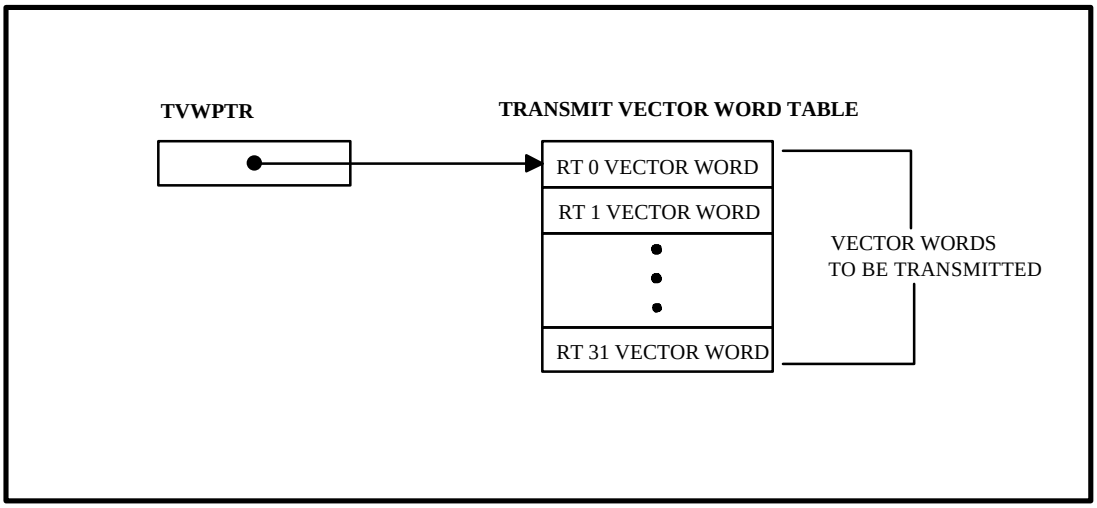


Figure 4-11 Transmit Vector Word Table

Last Sync Word Data Structure

The ABI microcode responds to a Synchronize with Data Word Mode Code command for a simulated RT by storing the data word from the received Mode Code command in the specific RT location of this 32-word table. The ABI updates this table regardless of whether the ABI is simulating or monitoring the RT (this is the only Mode Code structure which functions in Monitor Mode). The LSYPTR control register (offset 009Ah) holds the starting address for the table. The table provides storage for 32 Synchronize Data words, one for each RT. Figure 4-12 illustrates the table structure. The host software always has access to this structure in order to read the desired data word value. **This is a required data structure for ABI operations.**

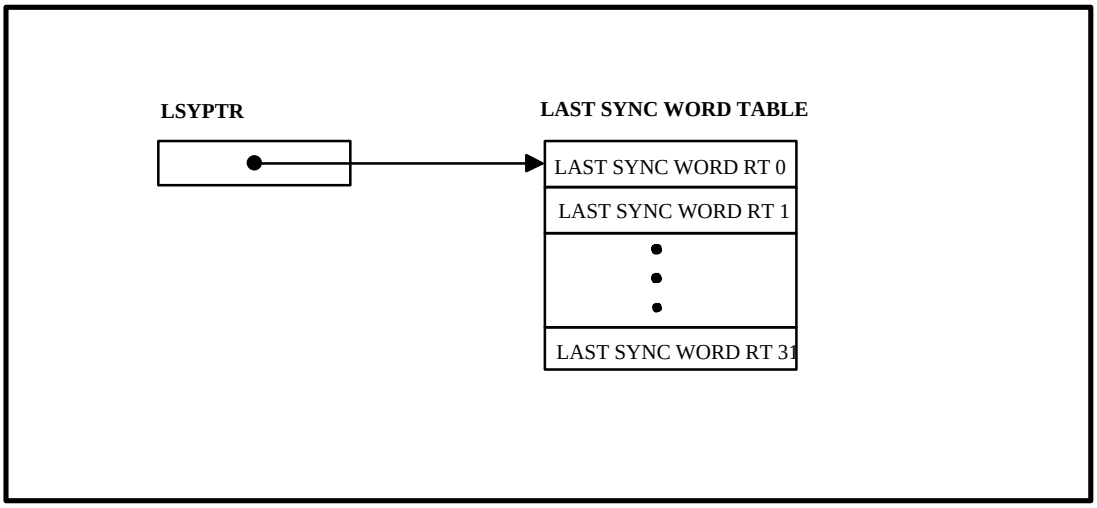


Figure 4-12 Last Sync Word Table

Last Command Data Structure

The last valid message the ABI received from a simulated RT is stored in this 32-word table. The table provides storage for the Last Command Word for each of the 32 Remote Terminals (The table position corresponds to the RT address). The ABI microcode responds to a Transmit Last Command Mode Code for a simulated RT with the last command word stored in the table. The LCDPTR control register (offset 0097h) holds the starting address. Figure 4-13 illustrates the table structure. The host should not access this structure. **This is a required data structure for ABI operations.**

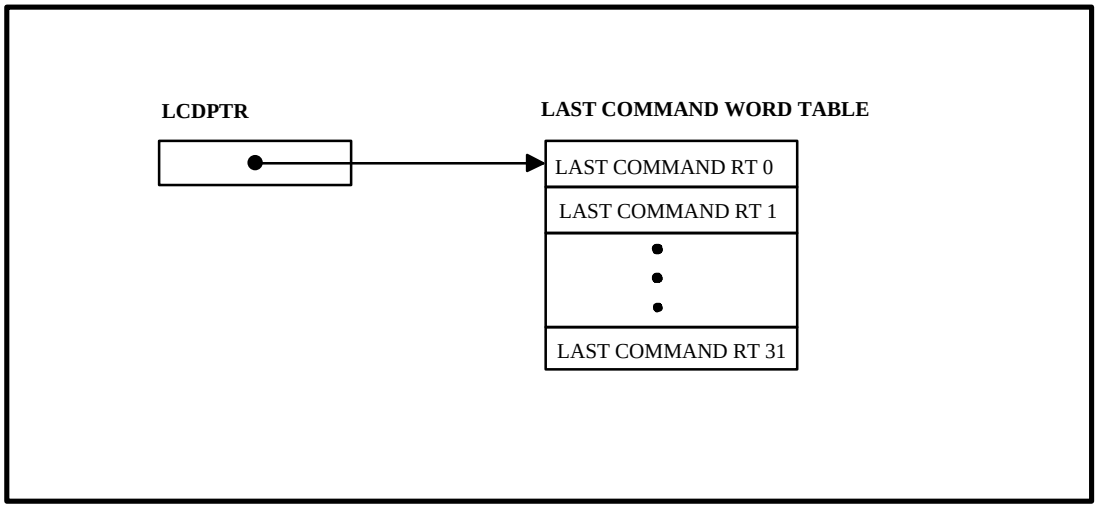


Figure 4-13 Last Command Word Table

Bit Word Data Structure

The host software is responsible for writing the Bit Word value to this 32-word table for each RT being emulated. The ABI microcode responds to a Transmit Bit Word Mode Code command for a simulated RT by returning the appropriate bit word from the table (The table position corresponds to the RT address). Figure 4-14 illustrates the table structure. The BITPTR control register (offset 0096h) holds the table starting address. **This is a required data structure for ABI operations.**

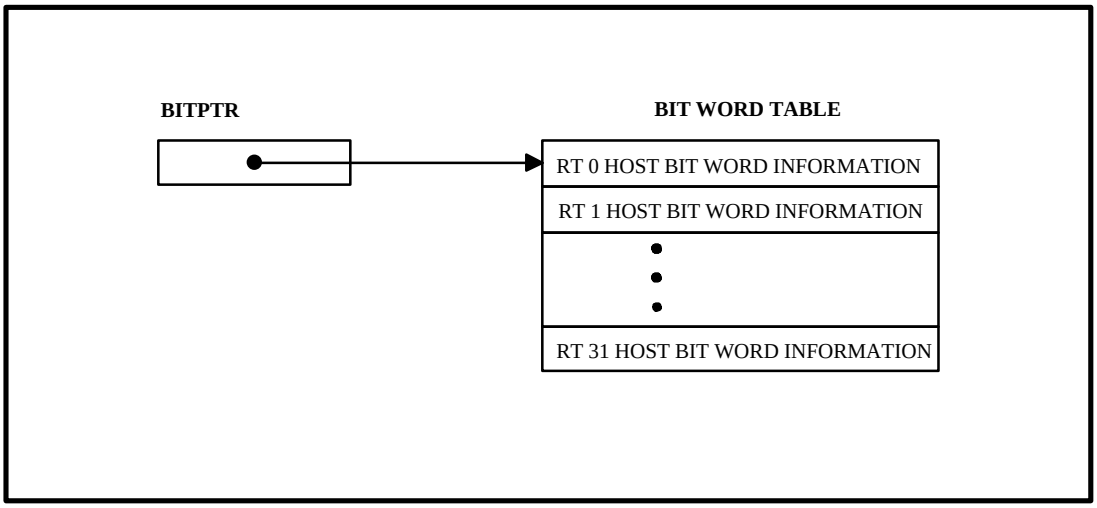


Figure 4-14 Bit Word Table

Optional Mode Code Data Structures

Filter Table

Provides an Interrupt upon the detection of a message received. See the Filter Table discussion of Bit 1 earlier in this subsection.

RT Address Data Structures

Provides a special data buffer format for Mode Codes that can be used to detect the occurrence of a Mode Code for specific Remote Terminals (See "RT Address Mode Code Data Buffer Format" below.).

Mode Command Interrupt Table

This structure may be used to set up Interrupts for specific Mode Codes (for specific Remote Terminals). It may be used in conjunction with or instead of the RT Address Data Structure for host detection of desired Mode Codes. This data structure provides a lower-level Interrupt capability for Mode Code detection than that provided by the Filter Table. (See "Mode Command Interrupt Table" description later in this subsection).

RT Address Mode Code Data Buffer Format

The Address Data Structure provides a special Mode Code format for Receive Subaddresses 0 and 31 (see Figure 4-15). This format allows the ABI microcode to store the number of times each Mode Code has occurred. It is available for all Mode Code commands detected by the ABI (including Mode Codes generated by ABI Bus Controller simulation). To utilize this structure, define a 34-word array data buffer (using the host software) for Receive Subaddresses 0 and 31, as illustrated in Figure 4-15. Set the Buffer Pointer Table Transmit Subaddresses 0 and 31 to NULL. (Only the data buffers for Receive Subaddresses 0 and 31 are used to store Mode Code occurrence counter values, regardless of the state of the T/R bit in the Mode Code command). Set the Pointer in each of these special data buffers so it points back to itself, creating a single-link data buffer list.

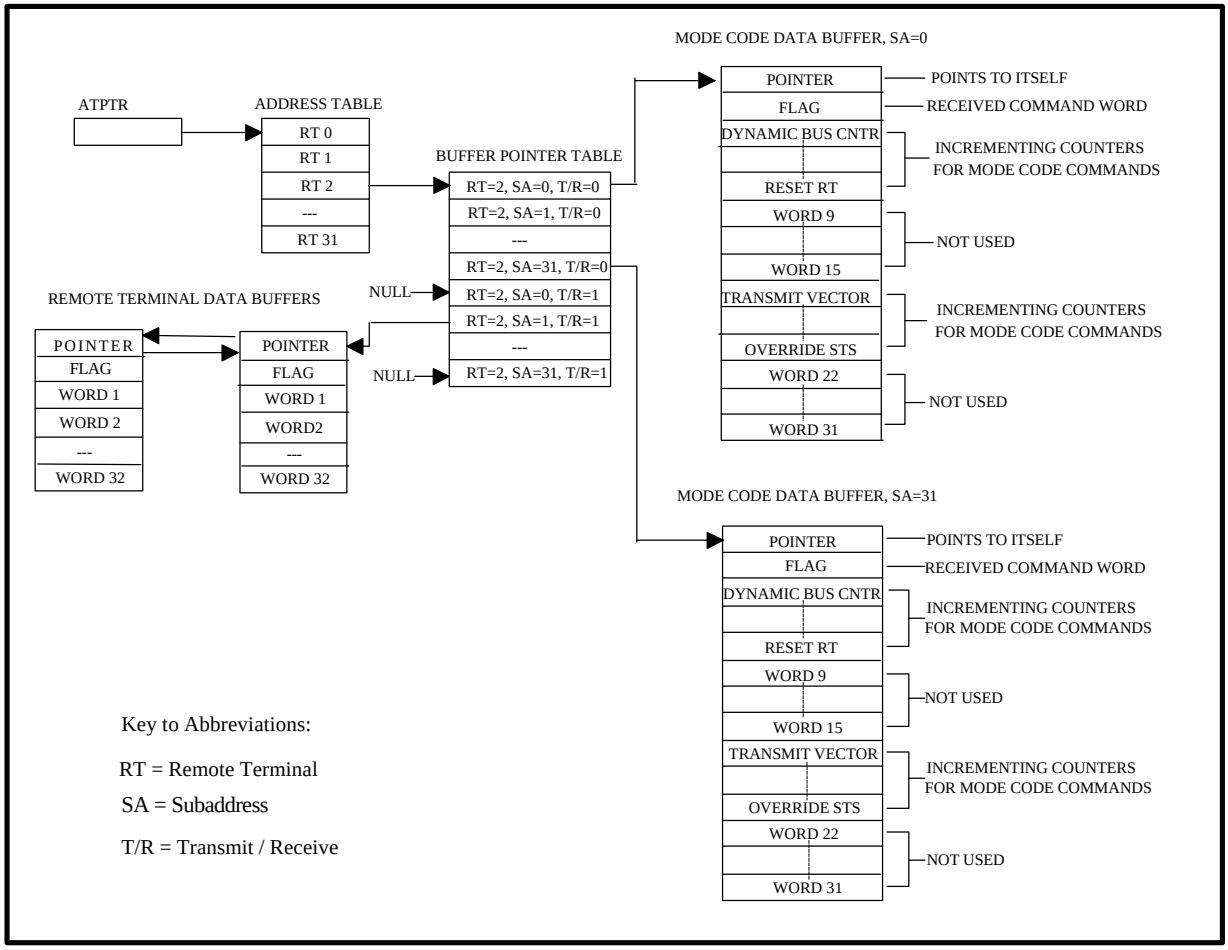


Figure 4-15 RT Address Mode Code Data Buffer

The ABI microcode increments the word value in this buffer by using the Word Count/Mode Code value as an offset to the data word area. For example, if a Transmit Vector Word Mode Code (value of 10000h or 16) is received for RT 9, Subaddress 31, the 19th word of the data buffer for RT 9 is incremented by "1". The received Command Word is stored in the flag word of the buffer.

This same structure may be used for Map Monitoring functions. See the Protocol Table (Figure 4-8) for information on enabling Map Monitor functions for specific Remote Terminals, and the "Bus Monitoring" subsection for an overview of monitoring Mode Code messages.

Mode Interrupt Mask Table

The MIMPTR control register (offset 009Eh) holds the starting address of the Mode Interrupt Mask Table. This table allows the host software to select Mode Codes (for each RT) that will cause a hardware Interrupt. *The ABI microcode will only generate Interrupts from this table if the corresponding RT is being simulated*. The ABI microcode checks for the appropriate Mode Code Mask bit to be set in this table when a particular Mode Code command is received. If the bit is set to "1", the ABI microcode generates a hardware Interrupt to the host software.

The table structure, illustrated in Figure 4-16, consists of a 64-word array, 2-word mask field for each RT, with the mask words arranged by RT address number. The first mask word for each RT provides bit selections for Mode Codes 00h to 08h (Dynamic Bus Control to Reset RT). The second mask word provides bit selection for Mode Codes 10h to 15h (Transmit Vector Word to Override Selected Transmitter Shutdown). See the "Interrupts" subsection for more details on ABI Interrupt handling.

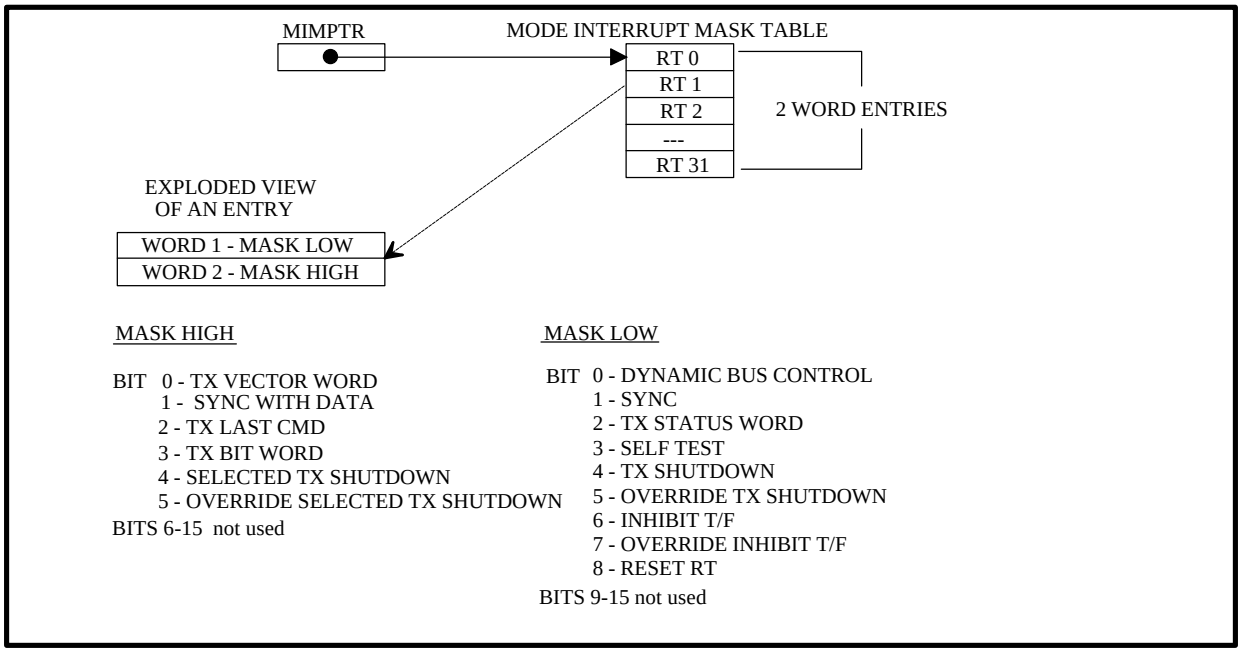


Figure 4-16 Mode Interrupt Mask Table

Variable Intermessage and Status Response Gap Times

BC Intermessage and RT Status Response Gap Times are variable through the use of ABI control registers **BCIGP** (offset AEh), **RSPGPA** (offset 00ACh) and **RSPGPS** (offset 00ADh). BCIGP determines the intermessage gap time for BC messages. RSPGPA determines the maximum gap time allowed by a Bus Controller or Remote Terminal waiting for an RT status response. RSPGPS determines the actual gap time a Remote Terminal waits before transmitting a status response.

To determine the value to be entered in each of these control registers, use the following formulas:

RSPGPS, BCIGP = (time_in_microseconds - 4.0) * gap_count

RSPGPA = (time_in_microseconds - 2.0) * gap_count

where time_in_microseconds is the desired gap time (for RSPGPS and BCIGP, must be greater than four; for RSPGPA, must be greater than two) and gap_count is the MIP rating of the ABI (see "BUFPTR" description in the "BC Simulation" subsection of this manual). All values are in decimal; convert the Gap Time to hexadecimal before entering it into the control registers. This value will be accurate to within five microseconds of the desired gap time.

The actual bus gap time is dependent on the type of message being sent. Mode codes may have longer response times than do normal status messages and errors will cause "wait timeouts" because the Remote Terminal will not respond with a status word.

1553 BUS MONITORING

The ABI can perform two independent modes of bus monitoring, Sequential Monitoring and Map Monitoring, while performing BC and RT simulation tasks. This subsection of the manual details each of these bus monitoring modes.

Sequential Monitoring stores raw 1553 traffic in double ("ping-pong") buffers for real-time data logging, recording and analysis applications. The ABI is capable of storing every 1553 message to double buffers and notifying the host system of buffer swaps through interrupt and polling functions. The messages are stored with 32-bit, 1- or 32-microsecond resolution time stamps. This monitoring method is useful when the host system has tight processing constraints (relative to 1553 traffic) but must still receive every 1553 message word. The key data structures for Sequential Monitoring are **Sequential Monitor Buffers**, the **Filter Table** and the **Interrupt Queue**.

Map Monitoring provides user defined linked-list data buffers for capturing RT commands and data words. This monitoring method allows the host system to set up fixed buffer locations for 1553 messages and then poll these buffers as host processing time permits. These data buffers may be chained together indefinitely, limited only by the available ABI memory. Map Monitoring is often used in conjunction with RT simulation applications where real-time polling of RT data buffers is preferred. The key data structures for Map Monitoring are **RT Address Table**, **Filter Table** and **Protocol Table**. The details of these structures may be found in the "RT Simulation" subsection.

Sequential Monitoring

Sequential Monitoring provides the host system a method of receiving every 1553 message *in the same data format as is presented on the bus*. The host system must define two equally sized buffers of any length (up to the maximum available ABI memory). It must also specify which RT and subaddress (SA) combinations will have their messages stored in these buffers. Messages will fill one buffer at a time. Switching to the second buffer will automatically occur only when the first is filled (when the second buffer is filled, the ABI will switch back to the first buffer to continue storage - the buffers will "ping-pong"). The ABI microcode switches buffers when there are less than 50 words remaining in the current buffer (the size of the largest 1553 RT-RT message plus overhead words). This action continues indefinitely, or until the host system disables this function.

Three structures must be defined before Sequential Monitoring can take place: 1) The Filter Table must be defined with the desired RT terminals programmed for traffic, 2) The Sequential Monitor Buffers must be defined and 3) The interrupt queue buffers must be defined. The starting addresses for all these data structures must be programmed in the Pointer Table. The following paragraphs as well as the "RT Simulation" subsection provide programming details for each of the above data structures.

Filter Table

The Filter Table provides programming options for RT Simulation and Sequential Monitor designation. Each of the possible 32 RT registers holds a pointer to its subaddress (SA) filter table. Each of the possible 64 SAs (32 receive and 32 transmit) has an integer flag word, used for different programming options. Each flag word contains a Sequential Monitor bit. **This bit indicates whether the messages for the respective RT/SA are to be placed in a Sequential Monitor buffer.** This function allows message filtering at the RT/SA address level. There are no limits on the number of RT/SA messages that can be filtered or stored to buffers. Please see the "RT Simulation" subsection for details about this structure (use the Table of Contents to locate the "Filter Table" discussion). Figure 4-17 is provided for reference.

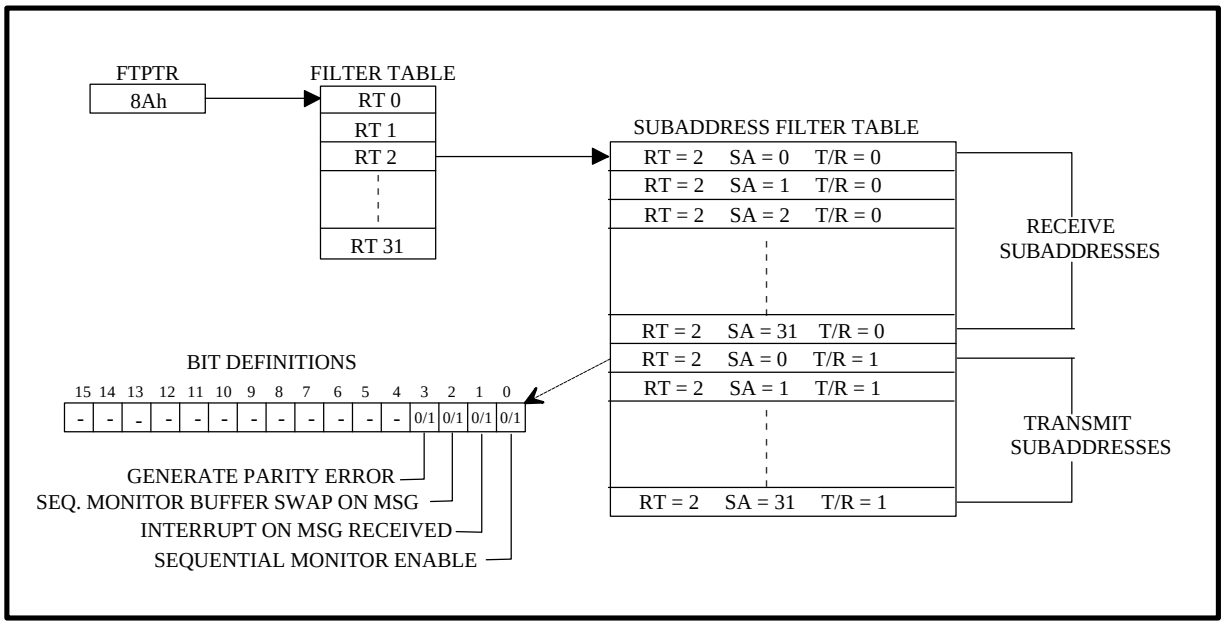


Figure 4-17 Filter Table Data Structures

Sequential Monitor Buffers

The Sequential Monitor Buffers provide a double buffered storage area for 1553 messages selected using the Filter Table. The two control registers, M1PTR (offset 8Ch) and M2PTR (offset 8Dh), provide the starting addresses of two message blocks in ABI memory (in the 0400h to FFFFh area). Both buffers must be the same word length and this length must be programmed into the MBLEN register (offset 008Eh). **It is not possible to disable the monitor function.** Sequential Monitor Buffers with minimum lengths of 50 (MBLEN >= 50) are a required data structure for proper ABI operations. The monitor buffer may generate a monitor buffer swap interrupt if this value is less than the default value.

The size of the buffers is determined by considering the host system's processing capabilities and the amount of 1553 bus loading. For example, considering the maximum transfer rate of the 1553 bus (about 49K words/second), two 25K buffers would be required. This would result in two buffer swaps per second (and probably two interrupts to the host system per second). Most 1553 buses have much lower transfer rates and most host systems can process much more than two interrupts per second; therefore, the actual buffer size required is much less (typical ABI setups include buffers of 4K-16K words).

If the sequential monitor bit is set to "1" for a particular RT/SA in the Filter Table, Sequential Monitoring will take place. When the ABI microcode detects a 1553 message from this RT/SA, it time stamps the message and stores it in the next available message block of the buffer pointed to by M1PTR. The microcode always uses the value in M1PTR as the starting address for the current buffer. When the current buffer is filled, the microcode swaps the values between M1PTR and M2PTR; therefore, M1PTR always points to the "current" buffer. (M1PTR is not updated with the current storage location within the buffer; this location is maintained internally by the microcode.) Figure 4-18 provides a description of the Sequential Monitor Buffer data structure.

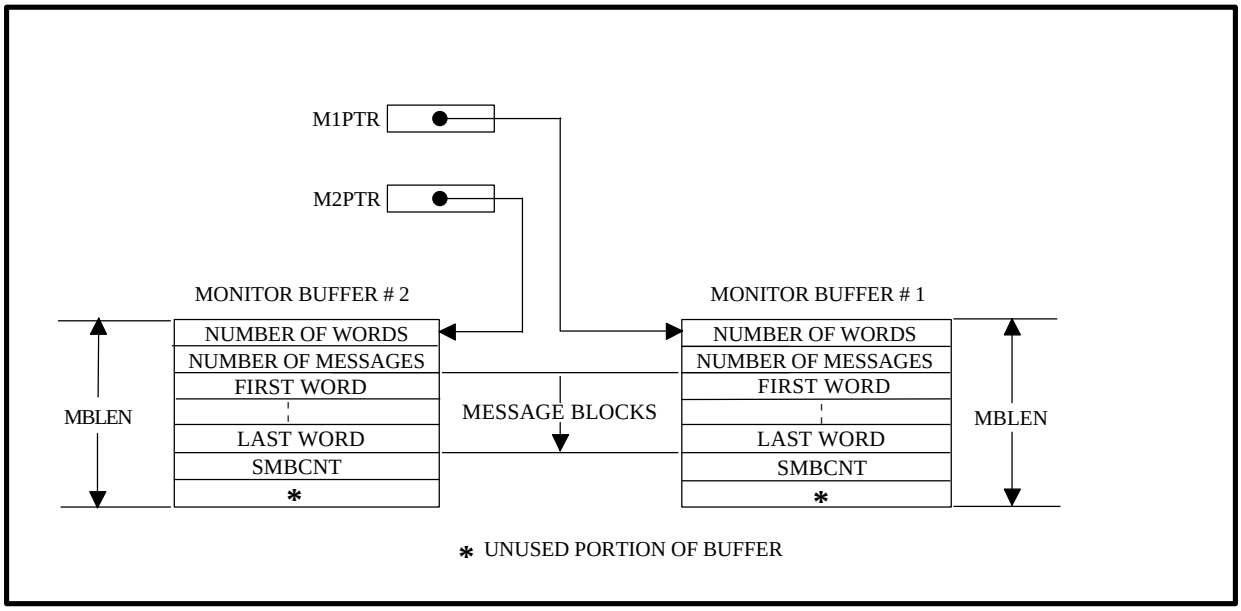


Figure 4-18 Sequential Monitor Buffer Data Structures

A message buffer counter is located at position "number of words + 1". It is initialized at "0" and, at the time of a buffer swap, is incremented once for each monitor buffer swap. It can be used by the host system to track message buffers and insure no buffer losses occurred. The word value stored in the Sequential Monitor Buffer Counter , SMBCNT (offset ABh), is used as the message counter in the subsequent buffer swap. The user may write a value to this register to preset this counter.

When a buffer has been filled by 1553 messages, the microcode updates words one and two of the buffer with the **total buffer word count**, and the **1553 message count**, respectively. These first two words are included in the word count.

The remainder of the buffer contains message blocks which consist of one **type** word and two time stamp words (SCHIGH and SCLOW), followed by the complete 1553 message as it appeared on the bus. An error word is stored after the bus data if an error occurred for the message. The message block structures are illustrated in Figure 4-19.

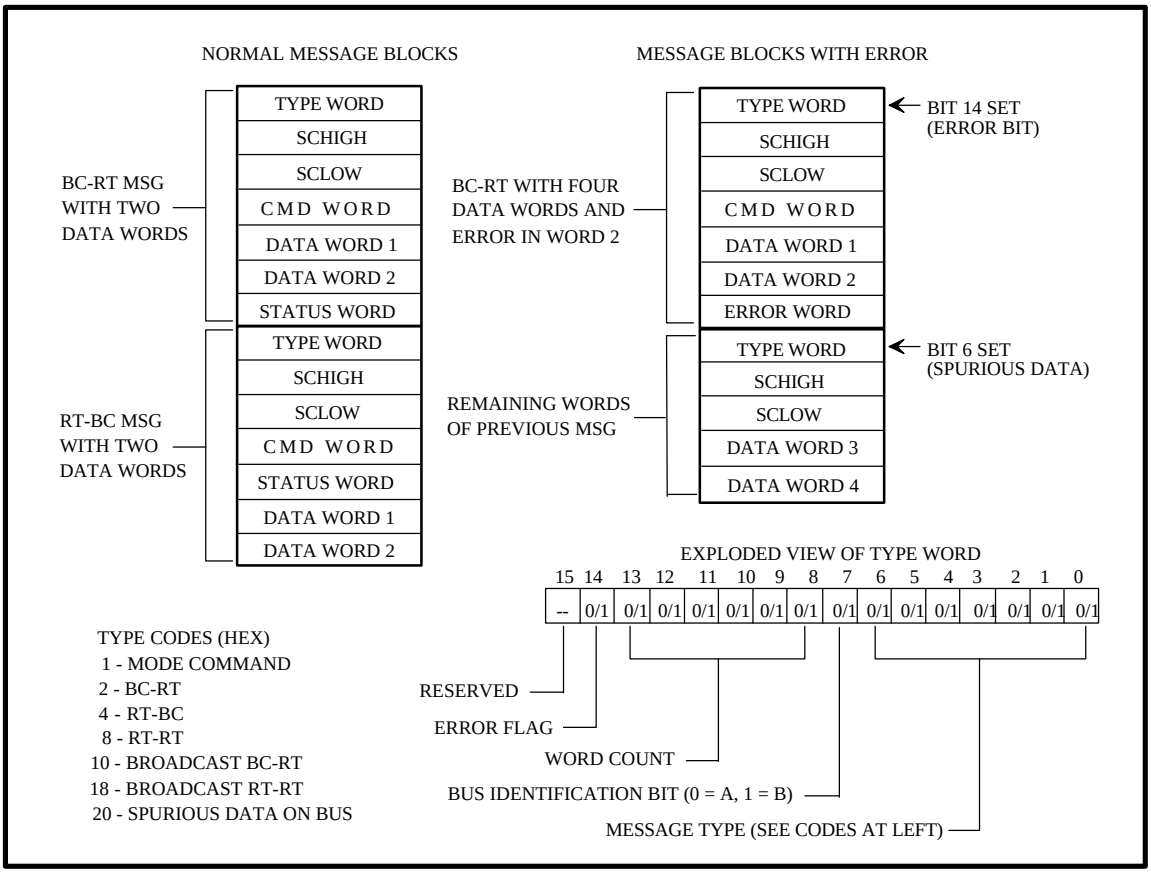


Figure 4-19 Sequential Monitor Buffer Message Blocks

The **type** word indicates the type of 1553 message that has been stored, the bus on which the message was detected, the word count of the message block (including the type word) and whether or not an error occurred. The host software can use this information to read the command word and determine the configuration of the status and data words (the word count field specifies the number of words in non-Mode Code messages, including the type, command, data and status words).

Bits 0-6 - Message Type
These bits define the type of 1553 transfer for the message.

Bit 7 - Bus Identification Bit
If this bit is "0", the message was received/transmitted on Bus A; if "1", the message was received/transmitted on Bus B.

Bits 8-13 - Word Count
These bits indicate the total word count of the message, including the Type, Time Stamp, Command, Data, Status, and Error words.

Bit 14 - Error Flag
If this bit is "1", the message contained an error; if "0", no error.

The **SCHIGH** and **SCLOW** words provide a 32 bit, 1 or 32 microsecond resolution value for message time stamping. SCLOW provides the least significant 16 bits of system clock and SCHIGH provides the most significant 16 bits. The clock may be set (or reset) through control I/O registers. (See "System Clock Registers" and "Timing Hardware Registers" in the "ABI Operations Overview" subsection of this *Full Function Programmer's Reference*.) The time stamp is relative to the beginning of the first command word of the message block.

The 1553 message for each block follows the SCHIGH and SCLOW time words. The 1553 data words are stored exactly as they appear on the 1553 bus. The format for BC-RT, RT-BC, RT-RT, Mode Code, and Broadcast Commands can be found in the 1553 standard.

If the ABI microcode detects an error during storage, it will add an error word to the end of the message block and set bit 14 of the type word to "1". The error word represents the address of an error in the Error Table (see Table 4-4 in the "ABI Operations Overview" subsection of this *Full Function Programmer's Reference)*. The error condition applies to the 1553 word immediately prior to the error word. For example, if a "parity error" is detected on the sixth word of a ten word BC-RT transfer, two message blocks will be generated. The first block will contain the command word and the first six data words. The error word will be 011Eh, the address for the "parity" error. The next message block will contain the remaining words of the message and bit 6 of the type word will be set (bit 6 designates spurious data).

The ABI microcode will place errors or spurious data into the buffers regardless of Filter Table bit settings. (Spurious data refers to unexpected data which appears on the bus.) This can cause unexpected buffer swaps if 1553 words appear that are of poor transmission quality. This condition can also be caused by error conditions such as RT "no response".

Minimum Programming Requirements for Sequential Monitor Buffers

Sequential Monitor buffers are a required data structure. MBPTR1 and MBPTR2 must point to buffers which are a minimum of 50 words in length (MBLEN must be at least 50 for each buffer).

Buffer Swap Detection

The user can detect when a buffer swap has occurred through hardware interrupts or by monitoring M1PTR and M2PTR for address changes. The ABI will force a hardware interrupt to the host system when a buffer swap occurs only if interrupts are enabled in the CSR control register (offset 0000h). Please see the "ABI Operations Overview" subsection discussion on Control I/O registers and the "Interrupts" subsection for more details.

The host system may detect a buffer swap by polling M1PTR for a change of address. Upon startup, M1PTR contains the address to the first sequential monitor buffer and M2PTR contains the address to the second buffer. When a buffer swap occurs, the address for M2PTR is copied to M1PTR and the M1PTR address value is copied to M2PTR. Thus, M1PTR always contains the address of the current buffer.

Forcing Buffer Swaps

The host software can force two types of buffer swaps: **simple buffer swap** and **buffer swap with interrupt**. Control register MBFLG (offset 009Fh) has two flags used to force these swaps. Write 0001h to the register to cause a simple buffer swap without an interrupt and write 0002h to the register to cause a buffer swap with a hardware interrupt (interrupt code 0002h). The ABI microcode will set the register to "0" when the swap has occurred.

Map Monitoring

Map Monitoring is typically used by data analysis systems or screen update applications. In these situations, the host computer must poll fixed locations in ABI memory for 1553 data at specific time intervals or on event interrupt. Special linked-list data buffers are set up by the host system to be used for this function. Data word buffers for any non-simulated RT can be indefinitely linked together, limited only by the available ABI memory. This allows the host system to configure the length of buffer lists as required by system resources. For example, if the host can only read data buffers 10 times a second, and the particular RT subaddress is updated 20 times a second, the linked-list should be two or three buffers.

Some ABI applications combine Map Monitoring with message interrupts. The host system first programs the Filter Table to produce a hardware interrupt when a critical message has occurred. The host then reads the Map Monitor data word buffer(s) for the respective RT during the interrupt service routine.

All of the Map Monitoring data structures are also used for RT simulation functions. A review of these data structures is included in this subsection with Figures 4-20 and 4-21 provided for reference. Detailed discussions are provided in the RT Simulation subsection.

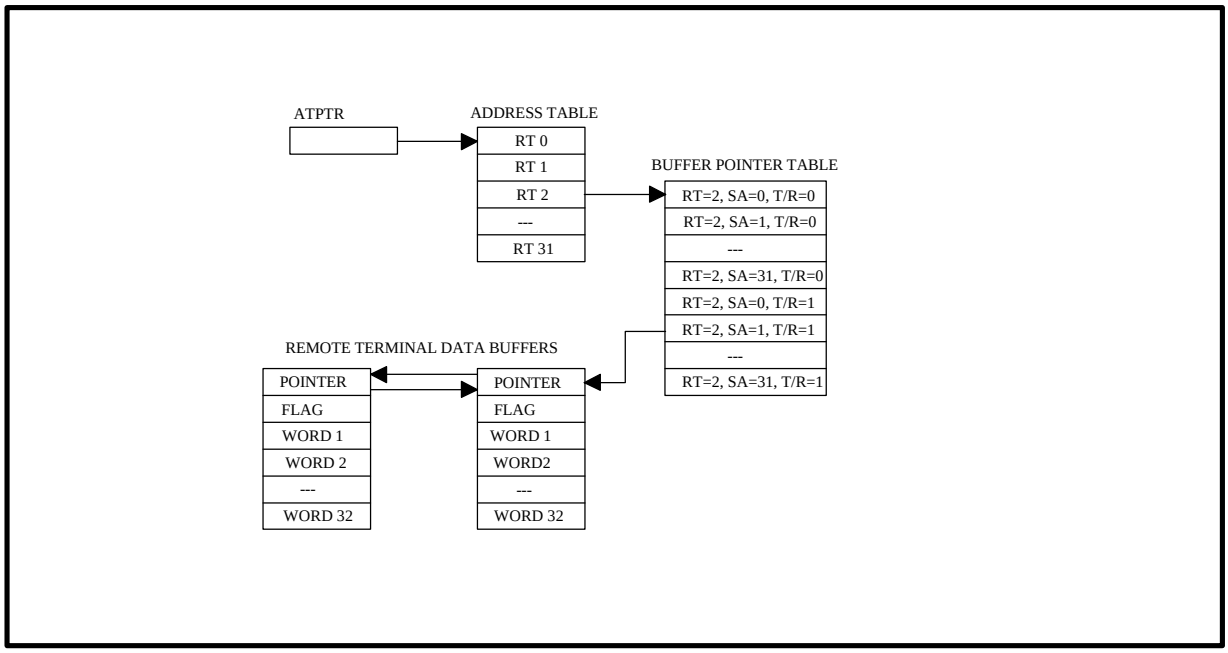


Figure 4-20 RT Address Data Structures

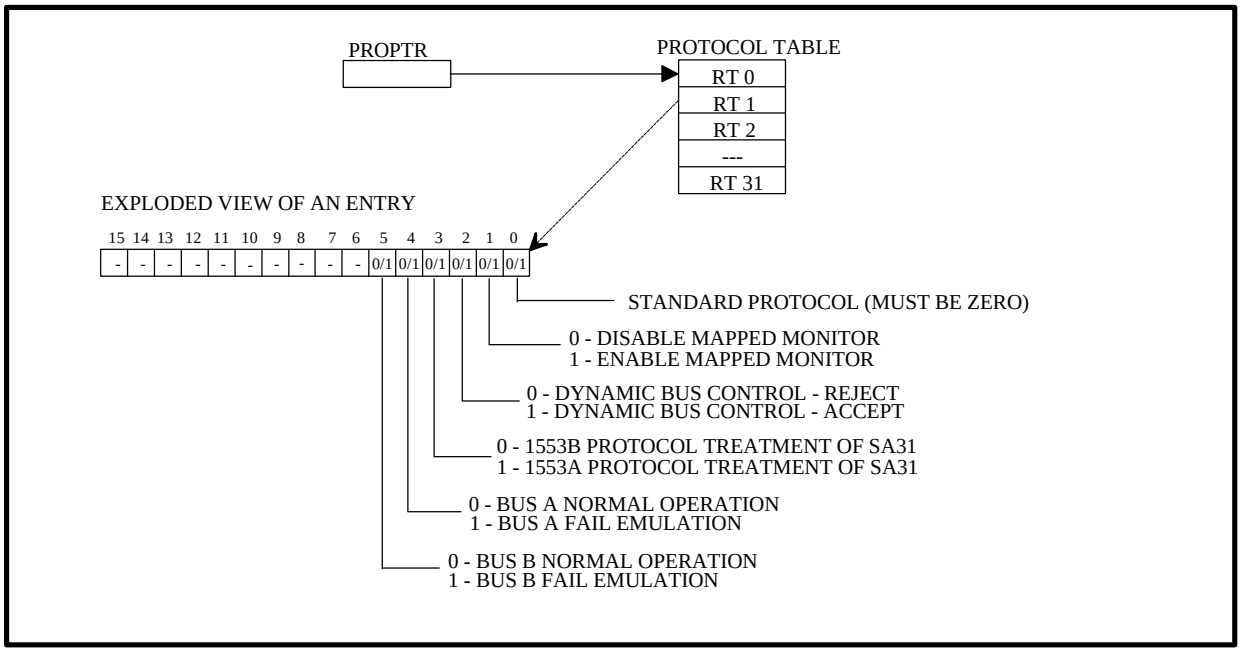


Figure 4-21 - Protocol Table

Map Monitoring Data Structures

The key data structures for Map Monitoring are the RT Address Table and the RT Protocol Table. The RT Address Table is the same structure used for RT Simulation data buffers. The ABI does not Map Monitor the RT if it is using the Address Table for simulation of the same RT.

The Protocol Table provides operation flags for ABI simulated RT responses, and provides a flag to enable and disable individual RT Map Monitoring functions. If Bit 1 is set to "0", the data words are not saved to RT Address Table Buffers. If Bit 1 is set to "one", the proper pointers and linked-list buffers must be established for the respective RT address, and subaddress-T/R bit combination. The ABI microcode first saves data to the buffer addressed by the Buffer Pointer Table, and then uses the pointer value in the data buffer link to determine the storage address for the next message.

Receive Broadcast Commands may be Mapped Monitored just as any other message. The RT Address Data Structures must be set up for RT address 31 with the appropriate subaddress data buffers defined.

INTERRUPTS

The ABI offers extensive interrupt capabilities for notifying the host processor of BC, RT, Monitoring and system events that have occurred on the 1553 bus. Interrupts may be enabled via the following structures: 1) Filter Table, 2) flag word of a BC block, 3) MBFLG, 4) BCSMSK, and 5) mode code interrupt data structures in MIMPTR.

The ABI utilizes a doubled buffered queue structure to store detected or generated events for the host system interrupt service routines. The queue addresses and event counters for this data structure are managed by control registers (See "Control Registers" in the "ABI Operations Overview" subsection). The host system reads these control registers during an interrupt service procedure to determine which interrupt queue is currently active and how many interrupt events are stored in the active queue.

The host system may handle interrupt events in one of two ways: 1) by polling the interrupt control registers with host software with the hardware interrupts disabled, 2) by using an interrupt service routine (ISR) written for the specific device with the hardware interrupts enabled. (SBS has developed many ISR routines for PC and UNIX systems which are available upon request to ABI customers.) The ABI fills the interrupt queues and updates interrupt control registers regardless of which method is used. If the host system does not require hardware interrupts, the **interrupt enable** bit in the CSR control register (offset 0000h) should be set to "0" at startup.

> **Note to VSB users:** See the Control/Status Register, bit 12, in the "ABI Operations Overview" subsection for more information on interrupts.

The interrupt queue data structures and control registers allow the host system to easily determine what type of BC, RT, Monitor or system event occurred. This subsection details the types of ABI interrupts and the data structures and control registers used by interrupt service procedures to manage interrupts.

Interrupt Types

There are interrupts associated with each major ABI function (BC and RT simulation and Monitoring) and interrupts for general system functions. Table 4-8 details the various ABI interrupt conditions. The "Code" column lists the numbers assigned to each interrupt condition. Each entry in an interrupt queue consists of four words; the first word representing the **code** type. The host system uses this code to determine how to handle each interrupt.

Table 4-8 ABI Interrupt Types

Type	Code (hex)	Description
BC Simulation		
RT Response Error	0005	By default, this interrupt is generated when an error is detected in an associated RT response. The most common error is RT "no response". Set bit 0 of a BC command block flag word to "1" to disable this interrupt.
Masked Status	0007	If bit 2 of a BC command block flag word is set to "1", the ABI will logically "AND" the associated RT status word with the control register BCSMSK (offset 0092h). If the result is greater than "0", the ABI will generate an interrupt. This interrupt is useful in determining when status word bits, other than address bits, are set to "1".
BC Halt	0006	By default, this interrupt is generated whenever BC execution is halted. This most commonly occurs due to a zero link in the last control block of a BC chain but an RT response error or an interrupt on masked status may also cause BC execution to halt. Set bit 3 of a BC command block flag word to "1" to disable this interrupt.
RT Simulation		
Command Word Detected	0003	This interrupt is generated if bit 1 of the Filter Table for a specific RT address and subaddress is set to "1", and the command word to the respective RT is received. This function is extremely useful for RT simulation and monitoring in determining when important command words have been received on the 1553 bus. The interrupt occurs after the completion of the entire BC message.
Mode Code Command Word Detected	0009	This interrupt is generated when a Mode Code command word is detected on the 1553 bus and the respective bit has been set to "1" in the Mode Command Interrupt Table.
Monitoring		
Sequential Monitor Buffer Swap	0002	This interrupt is generated when a monitor buffer swap occurs. A monitor buffer swap can occur during normal bus monitoring or via a forced buffer swap (write a "2" to the MBFLG control register 9Fh to force a buffer swap). The microcode stores 1553 messages, including messages with errors, according to Filter Table programming. (Spurious data on the bus is always stored in the Sequential Monitor buffer regardless of Filter Table programming.) This interrupt is frequently used by the host system to read one monitor buffer while the other buffer is filling with data.
Sequential Monitor Buffer Swap on Message	0010	The ABI generates this interrupt when bit 2 of the Filter Table for a specific RT/SA is set to "1" and the command word is received. This interrupt forces the sequential monitor buffer s to swap. The interrupt is generated after the message has completed and the monitor buffers have swapped.
Message Received	0003	This interrupt is generated if bit 1 of the Filter Table for a specific RT/SA is set to "1", and a command word for the RT/SA combination is detected.

Table 4-8 ABI Interrupt Types		
Type	Code (hex)	Description
General System		
Timer Data Ready	0001	The ABI generates this interrupt when control registers SCHIGH and SCLOW (offsets 0093h and 0094h) contain valid time words. This condition exists when the host software has set bit 0 of the control register CCW (offset 009D) to "1". Bit 0 of CCW is cleared by the ABI microcode when the interrupt occurs.
Interrupt Overflow	0008	This interrupt indicates that the interrupt queue has overflowed. The host system neglected to service the first interrupt of the queue before the entire queue filled with data.

Interrupt Queue Data Structures and Control Registers

The ABI manages interrupts using double buffered queues and counter control registers. When an interrupt event occurs, the ABI stores four words in the current interrupt queue. These four words consist of an interrupt code and three information words. The host reads these words to determine what type of interrupt occurred and to obtain related information such as the 1553 command words, pointer values and time values associated with the interrupt event.

The interrupt queues and control registers are illustrated in Figure 4-22.

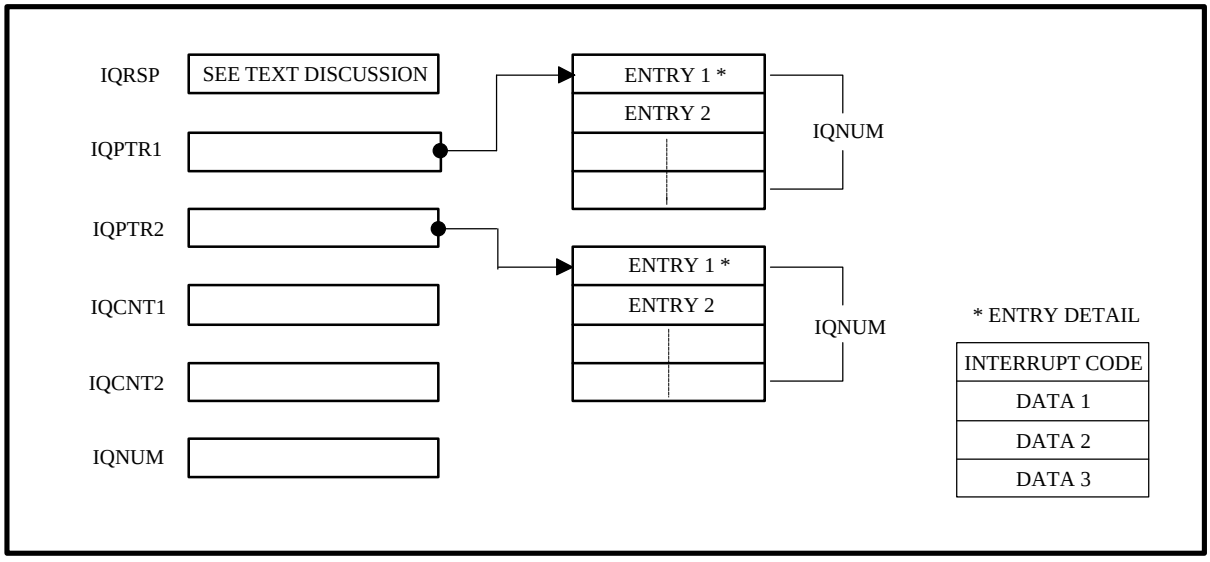


Figure 4-22 ABI Interrupt Queues and Control Registers

Minimum Programming Requirements for Interrupt Queues

The host software must program two interrupt queues. These queues must be identical in length and must hold a minimum of four interrupt entries each. A pointer (IQPTR1 and IQPTR2) must be programmed for each queue to indicate the starting address. Each queue must also have a counter (IQCNT1 and IQCNT2) to count interrupt entries as they occur. Each entry will consist of one to three words of valid data.

The host programs the starting address of the first queue into IQPTR1 (offset 0083h) and the address of the second queue into IQPTR2 (offset 0084h). The host programs the length of the queues (these lengths must be identical) into control register IQNUM (offset 0087h). The value in IQNUM determines the number of entries in each queue; each entry contains one to four words. For example, enter a value of "10" ("000Ah") into IQNUM for a 40-word queue. Ten interrupt entries may be stored in a queue this size. The ABI microcode defaults IQNUM to "4" at startup.

When an interrupt event occurs, the microcode sets the interrupt pending bit of the CSR control register (bit 7 of offset 0000h). If bit 3 of the CSR is set to "1" the ABI generates a hardware interrupt. The host ISR writes the value "FFFF" to IQRSP when starting the service routine and writes "0001h" to IQRSP when the routine is complete. The control registers IQCNT1 and IQCNT2 (offset 0085h and 0086h) provide an event count for each buffer (there are four words per event).

After microcode startup, the microcode writes interrupt events to the queue indicated by IQPTR1. When an interrupt occurs, the host ISR sets IQRSP to FFFFh. **The host must then wait two microseconds** for the microcode to finish processing any partially completed asynchronous interrupt events (see NOTE in box). If there is a partial interrupt pending, the microcode finishes processing, places the event information in the queue indicated by IQPTR1, and increments IQCNT1. If a new interrupt occurs during the host service (after FFFFh is written to IQRSP), the new interrupt event information is placed in the queue indicated by IQPTR2. After servicing the interrupts, the host ISR writes a "1" to IQRSP and the microcode swaps the values from IQPTR2 to IQPTR1 and from IQCNT2 to IQCNT1. If an interrupt event was placed in IQPTR2, the ABI will generate a new hardware interrupt.

> NOTE: If the host has programmed the ABI for interrupt events that are known to be synchronous, and these events occur more than 10 microseconds apart, the host ISR does not need to wait two microseconds before servicing the interrupts.

When the host services an interrupt, it reads the **code** word from the interrupt entry to determine the interrupt type. Some interrupt types consist of only this code word and others also contain data words. Table 4-9 defines each of the code words and their associated data words. Some words are undefined (their values are unknown) and should be ignored by the host software.

Table 4-9 Interrupt Word Descriptions				
Type	Word 1 - Code (hex)	Word 2	Word 3	Word 4
Timer Data Ready	1	Low 16 bits of system timer.	High 16 bits of system timer.	Undefined.
Sequential Monitor Buffer Swap	2	Undefined	Undefined	Undefined
Message Received	3	This command word for the respective RT Filter Bit for BC-RT or RT-BC transfers. For RT-RT transfers, this word contains the "receive command" word.	Set to "0" for BC-RT and RT-BC transfers. For RT-RT transfers, this word contains the "transmit command" word.	Undefined
RESERVED	4	Undefined	Undefined	Undefined
RT Transmit Error	5	The offset address for the BC command block from which the error was detected. Copied from BCCPTR pointer value at time of BC transmission. This usually indicates an RT transmission error or an illegal BC command block program setting.	The error code as listed in the error table. - See the "Error Table" in the "ABI Overview" subsection.	Undefined
BC Halt	6	Undefined	Undefined	Undefined
Masked Status	7	The offset address for the BC command block from which the RT status word "ANDed" with BCSMSK had a result greater than "0".	Undefined	Undefined
Interrupt Overflow	8	Undefined	Undefined	Undefined
Mode Code Command Word Detected	9	The mode code command word for the selected bit in the MIM table.	Undefined	Undefined
Sequential Monitor Buffer Swap on Message	10	The command word detected for the selected bit in the Filter Table.	Undefined	Undefined

Hardware Interrupt Service Procedure

The procedure shown in Table 4-10 is recommended for servicing ABI hardware interrupts.

Table 4-10 Hardware Interrupt Service Procedure

Step Number	Description
1	Host receives ABI hardware interrupt and starts ISR.
2	Host ISR reads IQRSP (offset 0082h) in a loop until IQRSP is equal to "0".
2	Host ISR writes "FFFF" to IQRSP.
3	Host ISR *waits 2 microseconds* for the microcode to swap values between IQPTRs and IQCNTs.
4	Host ISR reads IQCNT1 (offset 0085h) to determine the number of interrupt events to process.
5	Host ISR reads IQPTR1 to get the address of the interrupt queue to process.
6	The host reads each interrupt event, beginning at the address indicated by IQPTR1. The host reads each interrupt event in the queue, deciphering the code word to determine if data words are present. The host ISR may either process the interrupt data at this point or after execution has returned to the host software.
7*	Clear the pending interrupt bit. This step is performed by the hardware for VME systems.
8	Host ISR writes 0001h to IQRSP.
* Step 7 applies only to PC systems	

Software Polling Interrupt Service Procedure

The procedure shown in Table 4-11 is recommended for software polling of ABI interrupt events.

Table 4-11 Software Polling Interrupt Service Procedure

Step Number	Description
1	Host polls (reads) bit 7 of the CSR (offset 0000h) until it detects a value of "1". A "1" indicates an interrupt event has occurred on the ABI.
2	Host ISR reads IQRSP (offset 0082h) in a loop until IQRSP is equal to "0".
2	Host ISR writes FFFFh IQRSP.
3	Host ISR *waits 2 microseconds* for the microcode to swap values between IQPTRs and IQCNTs.
4	Host ISR reads IQCNT1 (offset 0085h) to determine the number of interrupt events to process.
5	Host ISR reads IQPTR1 to get the address of the interrupt queue buffer to process.
6	The host reads each interrupt event, beginning at the address indicated by IQPTR1. The host reads each interrupt event in the queue, deciphering the code word to determine if data words are present. The host ISR may either process the interrupt data at this point or after execution has returned to the host software.
7	Host ISR writes a "1" to bit 7 of the CSR control register to clear the interrupt request.
8	Host ISR writes 0001h to IQRSP.

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