Application Note AN-013

Moving from SBS to 1553 Products

abaco SYSTEMS

Copyrights

Abaco Systems, Inc.
26 Castilian Drive, Suite B
Goleta, CA 93117
Main +1 805-965-8000 or +1 805-883-6101
Support +1 805-965-8000 or +1 805-883-6097

support@abaco.com (email)

https://www.abaco.com/products/avionics

Additional Resources

For more information, please visit the Abaco Systems website at:

www.abaco.com

Contents

Introduction

This Application Note is intended for users that are familiar with the SBS 1553 products that need to move to the Abaco Systems 1553 products. This document will describe the significant differences between these product families and explain how these differences apply to 1553 applications. This provides a top-level overview of the differences – detailed information can be found in the SBS and Abaco Systems user's manuals and example programs.

This Application Note assumes a basic knowledge of 1553. For information on the MIL-STD-1553 protocol refer to the "MIL-STD-1553 Tutorial" document available from Abaco Systems, Inc. Additional information can also be found in MIL-HDBK-1553A.

SBS and Abaco Systems 1553 Product Families

Both SBS and Abaco Systems have several families of 1553 products. We will give a general overview of these product families and explain which of them are covered by this document.

SBS 1553 Product Families

"DSP-based" or "Second Generation" Products

The primary set of SBS 1553 products uses DSP processors and is called the "DSP-based" family of 1553 products. These products are available as single-function (ASF) or multiple-function (ABI) versions. This family includes the ASF/ABI-PCI, ASF/ABI-PMC/PMC2, ASF/ABI-PCMCIA2, ASF/ABI-PCMCIA2, ASF/ABI-cPCI3U, ASF/ABI-PC104, and ASF/ABI-V6.

This product family is the primary set of SBS products addressed in this document for porting to Abaco Systems products.

"FPGA-based" or "Third Generation" Products

SBS developed the "third generation" 1553 products to provide capabilities similar to our UCA 1553 products. This family includes the 1553-PCI3, 1553-PMC3, and 1553-3CP3.

The SBS "third generation" API functions only support Windows 2000/XP. These products are only available in PCI, PMC, and Compact PCI.

Porting these products to Abaco Systems should be very similar to the "DSP-based" products.

ABI-V5 and ABI-V7 Products

The ABI-V5 was the "high-end" VME 1553 product for SBS. It included features like error injection, RT illegal command option, and VMEbus DMA bus master capability. Special firmware loads were available for B2, Space Station, etc. This board is now obsolete.

The ABI-V7 was intended to replace the ABI-V5. It is a 6U VME single-board computer with a PowerPC processor and one or two PMC daughter-boards. It is available with 1553 or 3910 daughter-boards.

These products can be ported to our products but this document does not specifically address them differently than the "DSP-based" products.

Abaco Systems 1553 Product Families

"UCA" (Universal Core Architecture) Products

The original set of our 1553 products is called the "UCA" or "Universal Core Architecture" family of products. This family includes the R15-USB, QPCX-1553, QPCI-15553, QCP-1553, QPMC-1553, QPM-1553, QVME-1553, RQVME2-1553, Q104-1553-P, PCCARD-D1553, R15-EC, RXMC-1553, RXMC2-01553, RPCIe-1553, RAR15-XMC-XT/IT/FIO, R15-PMC R15_LPCIe and R15-MPCIe products.

This product family uses the BusTools/1553-API. Multiple-function versions of these products can be used with the BusTools/1553 Windows bus analyzer software.

This product family is the only set of our products addressed in this document for porting from SBS products.

"CORE and FlightCORE" Products

Abaco Systems developed the "CORE and FlightCORE" products to provide an FPGA IP-core product for embedded applications. The products have passed the 1553 RT Validation Test Plan. There are also board-level products that use this design. These products support both 1-Mbit 1553 and 10-Mbit MMSI protocols.

This product family uses the CORE-API.

This product family is NOT addressed in this document.

General Features

The following table lists the major differences in general features.

SBS	Abaco Systems
The SBS “DSP-based” 1553 products are available with ONE or TWO channels. The VME ABI/ASF-V6 is available with THREE or FOUR channels. The SBS “FPGA-based” products are available with up to FOUR channels.	Our UCA 1553 products are available with ONE, TWO, or FOUR channels.
SBS has a single API that includes functions to support many products (1553 DSP, 1553 FPGA, ABI-V7, ARINC, SDB, WMUX, DIO).	We provide separate APIs for different product families (UCA 1553, CORE 1553, CEI-x20, CEI-x30, etc.). The BusTools/1553-API is used for the UCA 1553 products.
The SBS analyzer (PASS-1000/3200) requires a different, more expensive version of the card.	Our analyzer (BusTools/1553) will run on any multi-function UCA 1553 board.
The SBS “DSP-based” 1553 products provide 128 Kilobytes of memory per channel. The SBS “FPGA-based” products provide 1 Megabyte per channel.	Our UCA 1553 products provide 1 Megabyte of memory per channel.
The SBS API provides 23 example programs for 1553, of which only 18 apply to the “DSP-based” products.	Our UCA 1553 API provides 72 example programs.
SBS only provides a SINGLE CHANNEL interface for PCMCIA.	We provide both SINGLE and DUAL CHANNEL interfaces for PCMCIA.

API Initialization – Device Information

The SBS API uses either a header file (dev_cfg.h) compiled with the application or a configuration file (sbs_dev.cfg) read at run-time to determine the board/channel that corresponds to each "device number". The API functions refer to this device number to select the appropriate board/channel.

```
/* Get the device number from the user. */
printf( "Enter the device number: " );
gets( str );
device_number = (SBS16)atoi( str );

/* Initialize the device. */
printf("Initializing . . . ");
if ( sbs_init_device( device_number,
                      4,
                      1000,
                      FROM_FILE ) == FAILURE )
{
   printf( "FAILURE!\n" );
   printf( "%s\n", sbs_read_error() );
   printf( ENTER_TO_EXIT );
   gets( str );

   return( SBS_1 );
}
printf( "SUCCESS.\n" );
```

Our API only requires that the user provide the device information in parameters to the initialization functions. We do not use a separate file for device configuration information.

```
// Constants to be used as parameters to
// BusTools_FindDevice and BusTools_API_OpenChannel.
// MODIFY TO MATCH YOUR CONFIGURATION.
#define MY_CARD_TYPE    PCCD1553
#define MY_INSTANCE     1
#define MY_CHANNEL      CHANNEL_1

BT_INT   dev_num, mode;

printf("Initializing API with BusTools_API_OpenChannel . . . ");

// First find the device based on type and instance.
dev_num = BusTools_FindDevice(MY_CARD_TYPE, MY_INSTANCE);
if (dev_num < 0) printf("ERROR IN BUSTOOLS_FINDDEVICE.\n");

// Open the device and get the channel id.
mode = API_B_MODE | API_SW_INTERRUPT;  // 1553B, use SW interrupts.
status = BusTools_API_OpenChannel( &ch_id, mode, dev_num, MY_CHANNEL);
```

API Handling of Single Function Devices

The SBS API provides a function to determine if a device is single function (m1553_is_asf) and a function to set the mode (m1553_set_mode). SBS requires that the user explicitly set the mode (BC, RT, or BM) for single function devices, and the user must assign an RT address for RT mode.

```
/* If it's an ASF card, you must set the mode. */
if ( m1553_is_asf( device_number ) )
{
   if ( m1553_set_mode( device_number, RT, 1 ) == FAILURE )
   {
      printf( "FAILURE!\n" );
      printf( "%s\n", sbs_read_error() );
      sbs_close_device( device_number );
      printf( ENTER_TO_EXIT_D );
      gets( str );

      return( SBS_1 );
   }
}
```

Abaco Systems does not require setting the single function mode. If the program attempts to initialize a second mode on a single function device the API will return an error. We do not require the user to assign a single RT address for RT mode because our single function devices can operate as multiple remote terminals (SBS only allows a single remote terminal).

Example Programs

Our BusTools/1553-API provides a large set of example programs (70+) that demonstrate many of the capabilities of the Abaco Systems 1553 products. These example programs are an excellent resource to use as a starting point for applications and users are encouraged to use these examples in their code. Many common questions can be answered by reviewing these example programs.

Bus Controller Features

The following table lists the major differences in Bus Controller Features.

SBS	Abaco Systems
SBS builds chains of messages. BC is started by chain number.	We build linked-lists of messages, where each msg has a “next” msg number. BC is started by initial msg number.
SBS – If the “loop flag” is set on the last msg in the chain, then the chain will repeat. Otherwise it will run once and stop.	Abaco Systems – Looping is determined by the next msg number. We provide a STOP block – use this for msg lists that run once and stop.
The SBS BC can run simple lists continuously without a defined frame time.	Ourr BC must define a frame time and must define beginning and end of frame messages.
SBS can have one or more buffers per BC message.	We support only a single data buffer per BC message. There is an option for two buffers, but these do not automatically swap, must be swapped by the software.
SBS does not provide BC conditional branching.	We provide BC conditional branching - CONDITION blocks.
SBS provides minor frame scheduling with start frame and repetition rate through SCHEDULE blocks.	We do not provide SCHEDULE blocks, but the same result can be achieved using CONDITION blocks with counter.
The SBS 1553 products do not support error injection. The ABI-V5/V7 do support error injection.	Our UCA 1553 products do support error injection.
SBS provides “BC DELAY” and “BC DUMP” block types.	We do not provide these block types. Delays between messages are done with the intermessage gap time (up to 64K microseconds). This will increase to 4M microseconds in the near future.

BC Example Programs

The SBS API provides the example program “test_bc.c” to demonstrate basic BC operation. This example sets up a BC list with two messages (1-R-1-32 and 1-T-2-32) with 8 buffers for each message.

Our example program “example_bc1.c” demonstrates basically the same thing. This example sets up a BC list with two messages (1-R-1-32 and 2-T-2-32) with 1 buffer for each message (the board does not support multiple BC buffers).

Remote Terminal Features

The following table lists the major differences in Remote Terminal Features.

SBS	Abaco Systems
The SBS 1553 products do not support the illegal command option. The ABI-V5 and V7 do support the illegal command option.	Our UCA 1553 products fully support the illegal command option down to the word-count level.
The SBS 1553 products enable RT response by subaddress. The RT only responds for subaddresses that have been defined, other subaddresses do not respond at all. This is not correct – if the RT is ON then it should respond to any valid command.	Our UCA 1553 products enable RT response at the RT level. The RT will respond to any valid command. The illegal command option can be used to identify subaddresses that are not implemented on the RT.
The SBS 1553 products do not support error injection. The ABI-V5 and V7 do support error injection.	Our UCA 1553 products do support error injection.
The SBS "DSP-based" 1553 products cannot simulate all RT addresses with all subaddresses because of the limited memory per channel.	Our UCA 1553 products can simulate all RT addresses with all subaddresses.
The SINGLE FUNCTION version of the SBS 1553 products will only simulate a SINGLE remote terminal.	The SINGLE FUNCTION version of our UCA 1553 products can simulate MULTIPLE remote terminals (31).

RT Example Programs

The SBS API provides the example program “test_rt.c” to demonstrate basic RT operation. This example sets up a Remote Terminal (RT 1) with two subaddresses (SA1 Receive and SA2 Transmit) with eight buffers per subaddress.

Our example program “example_rt1.c” demonstrates basically the same thing. This example sets up RT1 with SA1 Receive and SA2 Transmit with one buffer per subaddress. Our board does support multiple buffers per subaddress (defined in the RT SA control buffer) so this example could be modified to use eight buffers per subaddress if desired.

Bus Monitor Features

The following table lists the major differences in Bus Monitor Features.

SBS	Abaco Systems
SBS records messages into one of two large buffers (active and service buffer). Messages are read when buffers swap.	We record messages into a linked list of “message buffers”, where each buffer contains a single message. Messages are read at any time.
SBS buffers must be processed to extract the individual messages.	Our BM data is provided in a list of records with one record per message.
The SBS bus monitor can interrupt on buffer swap.	Our bus monitor can interrupt on every message. The BM can generate a TRIGGER OUT signal when it detects specific messages.
BM data from the API is not in the same format used by the analyzer (PASS-1000/3200).	BM data from the API is in the same format (.BMD) as used by the analyzer (BusTools/1553).
The PASS analyzer archive file formats are difficult to parse and process. The newer PASS-3200 analyzer uses a different format than the older PASS-1000 analyzer.	Our BM file format (.BMD) is very simple to process. We provide a conversion utility with source code to convert PASS-1000 archive files to the BMD file format.
The SBS BM is word-based. It will report errors on COMMAND words and will report spurious data.	Our BM is message-based. Without a valid command word there is no message. It does NOT report errors on command words and it does not report spurious data. Messages without a valid command word are not reported at all.

BM Example Programs

The SBS API provides the example program “test_sqm.c” to demonstrate basic BM operation. This example sets up a Bus Monitor with no filtering (captures everything), waits for one buffer to fill, and writes the captured messages to an ASCII file.

Our example program “example_bm1.c” demonstrates basically the same thing. This example sets up a Bus Monitor with no filtering (captures everything), prompts the user for the number of messages to capture, waits for the requested number of messages, and writes the messages to a BMD file. The BMD file can be viewed from the BusTools/1553 analyzer or converted to ASCII (by the BusTools/1553 analyzer software or by a standalone utility available from Abaco Systems with source code for no cost).

Bus Monitor Data Formats

The major issue for porting BM applications from SBS to Abaco Systems is the difference in the BM data formats and how the application processes the BM data.

The SBS library reference manual does not describe the BM data format for the SBS boards. The SBS API does include functions (m1553_output_bsm_buffer and m1553_t_output_bsm_buffer) that will convert buffers of BM data to an ASCII file. The source code for these functions can be used to see how SBS BM data is processed. SBS stores BM data in two large buffers (active and service buffers). The user application can read data from the inactive (service) buffer. The buffer must be processed to extract the individual messages in the buffer.

Abaco Systems application note AN009 (The BMD File Format) provides an explanation of the BM data format used by Abaco Systems. Each message is stored in a record structure (API_BM_MBUF structure). This makes it very easy to process BM data on a message by message basis.

Other Features

The following table lists differences in other features.

SBS	Abaco Systems
The SBS 1553 products do not support bus playback.	Our UCA 1553 products and API provide the same playback capability as the analyzer product (BusTools/1553).

SBS to Abaco Systems 1553 Product Matrix

The following table shows the correlation between SBS and Abaco Systems 1553 products.

Format	SBS	Abaco Systems
PCI	ASF/ABI-PCI 1553-PCI3	QPCI-1553 QPCX-1553
PMC	ASF/ABI-PMC, PMC2 1553-PMC3	QPMC-1553 QPM-1553
Compact PCI	ASF/ABI-cPCI3U 1553-3CP3	QCP-1553
PCMCIA	ASF/ABI-PCMCIA2	PCCARD-D1553
PC/104	ASF/ABI-PC104	Q104-1553
PC/104 *Plus*	*(N/A)*	Q104-1553-P
VME	ABI-V5 ASF/ABI-V6 ABI-V7	QVME-1553