Total COTS Solutions for Embedded 1553

White Paper By
Mike Glass
Technical Marketing Manager
Data Bus Products
Data Device Corporation

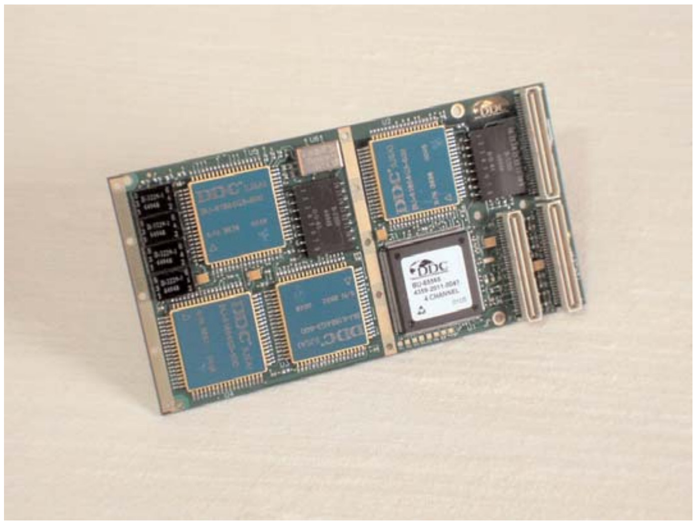

FIGURE 1. BU-65565 PMC CARD

INTRODUCTION

For contemporary avionics design, there's a need to strategically allocate engineering resources without increasing time to market. This environment has engendered an increasing reliance on Commercial-off-the-Shelf (COTS) hardware and software, with a demand for total solutions that may be leveraged over a range of applications. This entails the needs for ruggedized hardware, along with re-usable high-level software tools. The COTS paradigm includes a focus on cost effectiveness and high reliability, along with obsolescence and end-of-life issues for both cards and components.

While higher speed alternatives such as Fibre Channel loom, MIL-STD-1553 continues to be the workhorse network for military avionics integration. There are strategic benefits to be realized by mechanizing this interface in a portable, modular form factor such as a PCI Mezzanine (PMC) or PC/104 card. Such cards provide a convenient and flexible method for deploying 1553 on both COTS and custom-designed single board computers.

A major consideration is software development, which represents a significant cost element for embedded systems, the major issues being development cost and schedule. Related factors include portability, embedded code size, computational resources, and control over the development, validation, and documentation of flight critical and mission critical code. The availability of tools such as libraries, VxWorks drivers, and support of an offline development environment greatly enhance the usability of standardized hardware.

BU-65565 PMC CARD

DDC's BU-65565 card, illustrated in FIGURE 1, is a single width PMC card, built in accordance with the VITA 20-200X draft specification for conduction cooled PCI mezzanine cards (CCPMC). The BU-65565 includes one to four 1553 dual redundant channels, along with a 32-bit, 33 MHz PCI target interface, which can operate in either a 3.3 or 5 volt PCI signaling environment. The PCI interface includes a built-in FIFO for write transfers, which allows a 32-word transfer to be completed in about 1.5 µs.

Each 1553 channel is based on a BU-61864 Enhanced Mini-ACE. The Enhanced Mini-ACE architecture provides highly autonomous operation for BC, RT, and Monitor modes, along with 64K words of RAM per channel. The card is available with either conduction or convection cooling.

PMC CARD: THERMAL DESIGN

Referring to FIGURE 2, the thermal design of the conduction-cooled version of the PMC card includes thermal vias under the BU-61864's transceiver chips. The transceiver chips have the highest dissipation on the card: 1.22 watts max. at 100% transmit duty cycle. Thermally conductive epoxy in the form of a paste adhesive is applied to the PC board in the areas indicated by the highlighted (square) areas under the Enhanced Mini-ACE devices. Heat is conducted through the thermal vias to an inner copper plane layer, which functions as a heat spreader. The heat path includes additional thermal vias from the thermal plane layer to the two copper strips which run the width of the card. Thermal rails from the base card may then be bolted to the copper strips, providing a path for removing heat from the card.

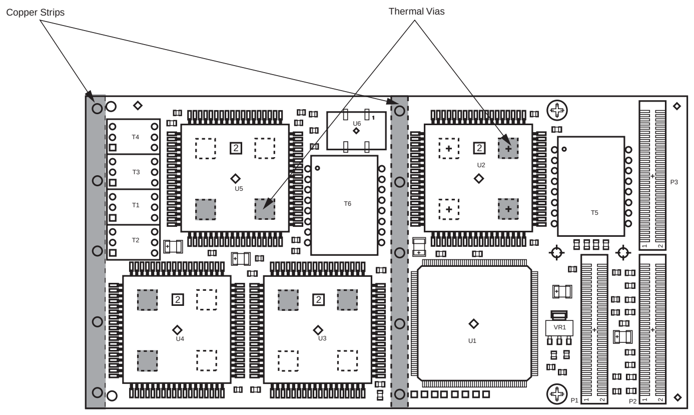

FIGURE 2. PMC CARD THERMAL DESIGN

The BU-65565MX card's total thermal resistance, from transceiver chip junction to the copper strip/thermal rail interface, is 39° C/W max. This includes the maximum θ_{JC} of 11° C/W for the BU-61864 hybrid, and a thermal resistance of 28° C/W for the card; i.e., from the hybrid case to the copper strip/rail interface. With a rail temperature of 85°C, this results in a maximum junction temperature of 133°C at 100% transmit duty cycle. Since the transmit duty cycle for most 1553 BCs and RTs is significantly below 100%, this provides ample headroom below the transceiver chip's maximum junction temperature of 160°C.

SHOCK AND VIBRATION TESTING

Shock and vibration tests were performed on the BU-65565 card. Following these tests, there was no evidence of physical damage. The shock testing consisted of a half-sine waveshape, with an input amplitude of 40 g's, duration of 11 ms, in each of six directions: horizontal (+X), horizontal (+Z), vertical (+Y), horizontal (-X), horizontal (-Z), vertical (-Y).

The vibration testing consisted of a random input of 15 to 2000 Hz, at an overall RMS level of 14.0 g's. The test duration was one hour applied independently to each of three orthogonal axes resulting in a total test time of 3 hours. Testing was performed along the Horizontal (X), Horizontal (Z), and Vertical (Y) axes.

BU-65567/8 PC/104 CARD

DDC's BU-65567/68 PC/104 card, illustrated in FIGURE 3, meets all requirements as specified in PC/104 Specification Version 2.3 for a 16-bit PC/104 module.

The BU-65567 version of the card provides between one and four MIL-STD-1553 dual redundant RT-only channels. Each channel includes a BU-61743 Enhanced Mini-ACE, which incorporates 4K X 16 of shared RAM.

The BU-65568 version of the card provides between one and four MIL-STD-1553 dual redundant BC/RT/Monitor channels. Each channel includes a BU-61864 Enhanced Mini-ACE, which incorporates 64K X 17 of shared RAM.

The BU-65567/8 card is available convection cooled. If necessary, it may also be supplied in a conduction cooled form factor.

PC/104 CARD: ADDRESSING MODES

In order to support operation for a variety of hardware/operating system platforms, the BU-65568 BC/RT/MT version of the PC/104 card may be jumper-configured to operate in either of two different addressing modes. The two addressing modes are: (1) a 32 Kbyte (16 Kword) paging mode; and (2) a flat addressing mode, which maps to an area of up to 512 Kbytes, to accommodate up to four Enhanced Mini-ACE channels, with each channel containing 64K X 17 (128K X 8) of register/RAM address space. The paging mode may be used on an Intel platform running DOS. For an environment such as a VxWorks running on a Motorola processor, the flat mode is more likely to be used.

ENHANCED MINI-ACE

The Enhanced Mini-ACE (FIGURE 4) provides highly autonomous operation for bus controller, remote terminal, and bus monitor modes. This includes multiprotocol support of MIL-STD-1553A/B, McAir, and STANAG-3838 for all three modes. The BU-61743 or BU-61864 Enhanced Mini-ACE integrates dual transceiver; along with protocol, host interface, memory management logic; and either 4K or 64K words of RAM.

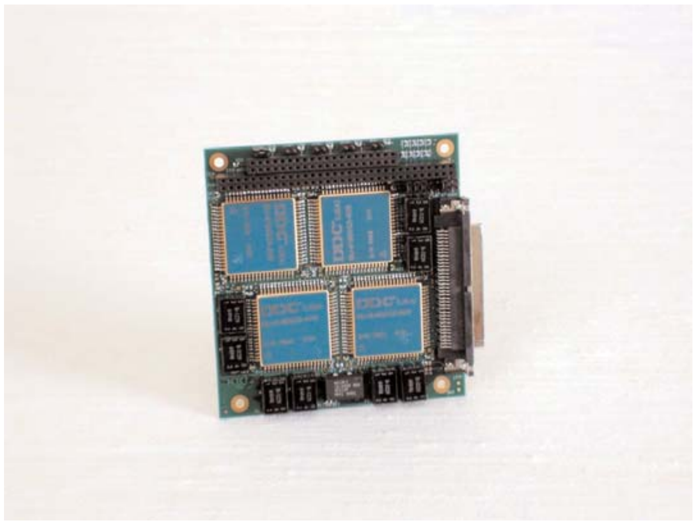

FIGURE 3. BU-65568 PC/104 CARD

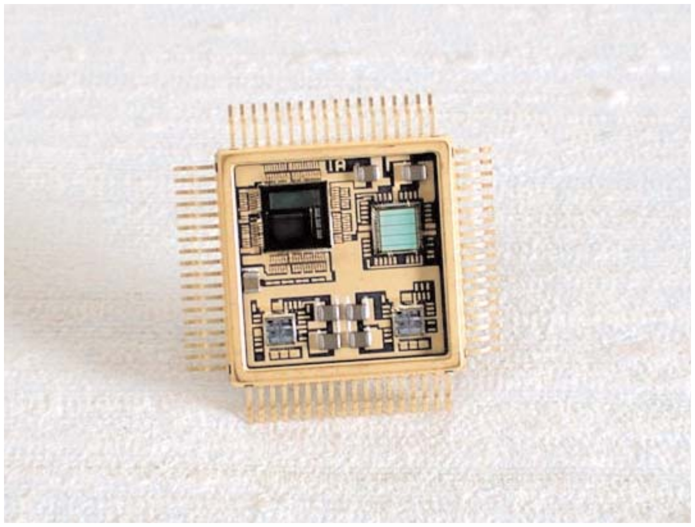

FIGURE 4. BU-61864 ENHANCED MINI-ACE

The Enhanced Mini-ACE's leading architectural features include a BC with a built-in message sequence control processor, an RT supporting multiple buffering modes, and a fully selective message monitor. These features offload the embedded host processor and software for a variety of applications.

BUS CONTROLLER

One of the salient features of the Enhanced Mini-ACE is its bus controller architecture. The enhanced BC's highly autonomous message sequence control engine offloads the host processor for such operations as minor and major frame scheduling, message retry and channel switching schemes, data double buffering, bulk data transfers, and asynchronous message insertion.

FIGURE 5 illustrates the Enhanced Mini-ACE BC message sequence control architecture. In operation, the BC's message sequence control engine fetches the op code and parameter word referenced by the current value of the BC Instruction List Pointer Register. The BC's primary instruction is Execute Message (XEQ). For this op code, the parameter is a pointer to the Message Control/Status Block.

The Control/Status Block is used for specifying a message's bus channel, message format, retry scheme (if any), 1553 command word(s), status word masking, and intermessage gap time. After a message has been processed, the host may read the message's block status word(s), time tag, and RT status words from the Control/Status block.

TABLE 1 illustrates the BC's 20-instruction set. As shown, each instruction includes an associated parameter. The parameter may be a Message Control/Status Block address, an Instruction List address, an interrupt bit pattern, a time value, a mechanism to modify general purpose flag bits, an immediate value, or a memory address. For seven instructions, the parameter is essentially a "don't care" (i.e., not used, shown as "—" in TABLE 1).

Most of the BC instructions are conditional. That is, it's possible to configure their execution to depend on the value of a particular condition code. Execution may be designated to be conditional based on the validity (or invalidity) of the most recent message, the value of a general purpose flag bit, or the result of a timer comparison operation. For an individual instance of an instruction, execution may be designated to be unconditional. In addition, it's possible to cause a particular op code in the instruction list to be skipped, in essence, making the instruction into a "no op". Note that four instructions are unconditional: Compare to Frame Timer (CFT), Compare to Message Timer (CMT), GP Flag Bits (FLG), and Execute and Flip (XQF).

The Enhanced Mini-ACE includes several mechanisms for communicating between the BC processor and the host processor: (1) the BC includes 8 general purpose flag bits. These may be used to operate as semaphores, enabling the host to communicate with the BC, and/or the BC to communicate with the host; (2) a general purpose queue. This is a 64-word circular buffer. The BC processor includes instructions to store block status words for specific messages, the current value of the time tag register, an immediate value, or an indirect value; (3) four user-defined interrupts. The latter may be used in conjunction with the flag bits or general purpose queue to alert the processor of specific data transfer events or error conditions, along with an option for a multi-subaddress global circular buffer.

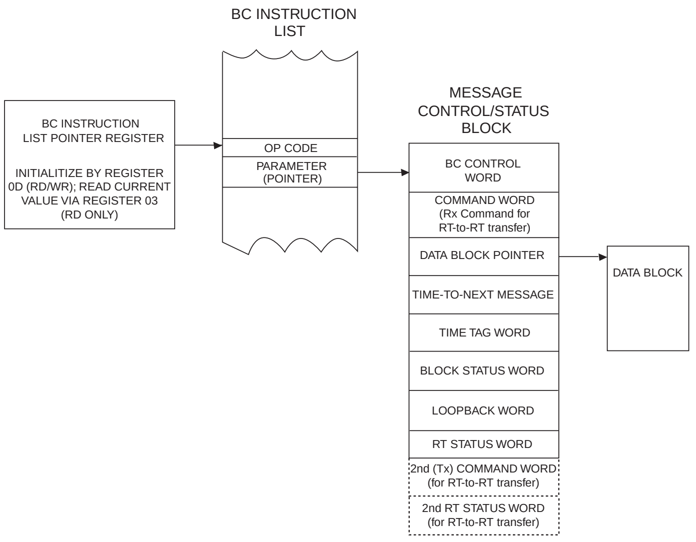
BC INSTRUCTION LIST
BC INSTRUCTION LIST POINTER REGISTER
INITIALITIZE BY REGISTER 0D (RD/WR); READ CURRENT VALUE VIA REGISTER 03 (RD ONLY)
OP CODE
PARAMETER (POINTER)
MESSAGE CONTROL/STATUS BLOCK
BC CONTROL WORD
COMMAND WORD (Rx Command for RT-to-RT transfer)
DATA BLOCK POINTER
TIME-TO-NEXT MESSAGE
TIME TAG WORD
BLOCK STATUS WORD
LOOPBACK WORD
RT STATUS WORD
2nd (Tx) COMMAND WORD (for RT-to-RT transfer)
2nd RT STATUS WORD (for RT-to-RT transfer)
DATA BLOCK

FIGURE 5. ENHANCED MINI-ACE BC MESSAGE SEQUENCE CONTROL

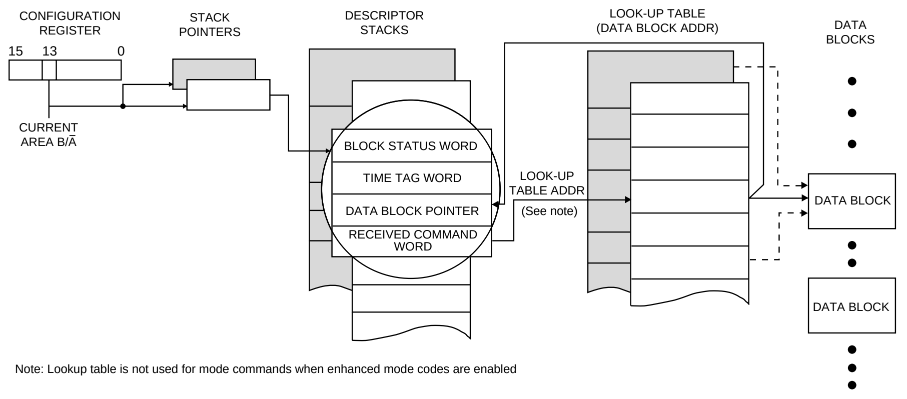

FIGURE 6A. RT SINGLE MESSAGE MODE

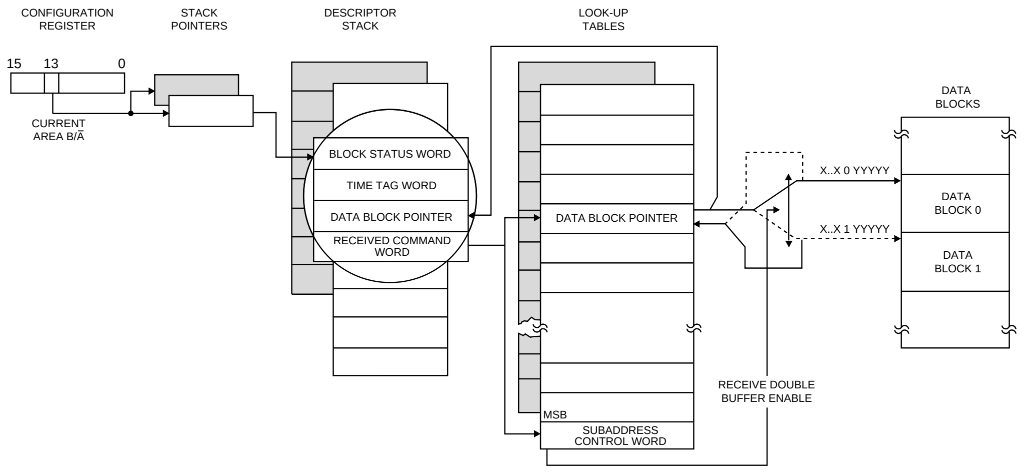

FIGURE 6B. RT DOUBLE BUFFERED MODE

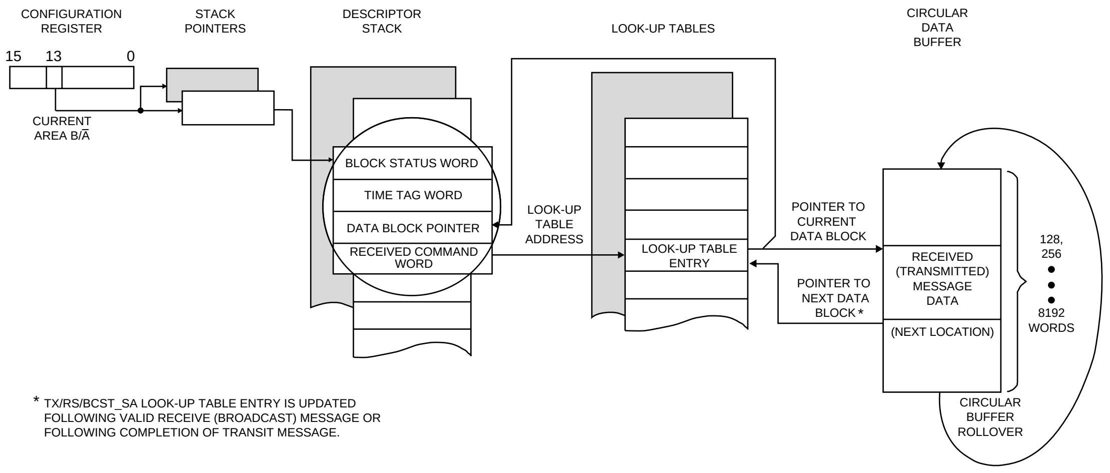

FIGURE 6C. RT CIRCULAR BUFFERED MODE

The global circular buffer option provides an option for a common buffer for multiple or all subaddresses. Other features of the RT architecture include a 50% rollover interrupt for circular buffers, an interrupt status queue for logging up to 32 interrupt events, and an option to automatically initialize to RT mode with the busy bit set following power-up.

FIGURE 6 illustrates the Enhanced Mini-ACE's three RT buffering modes. The single buffered mode (FIGURE 6a) provides a simple "mailbox" mechanism for allocating data blocks for individual subaddresses. The single buffered mode is applicable for receiving messages that trigger interrupts. In addition, by programming the host to toggle between a pair of data block addresses, the single buffered mode may also be used to ensure sample data consistency for transmit messages.

TABLE 1. ENHANCED MINI-ACE BC INSTRUCTION SET

INSTRUCTION	MNEMONIC	PARAMETER
Execute Message	XEQ	Message Control/Status Block Address
Jump	JMP	Instruction List Address
Subroutine Call	CAL	Instruction List Address
Subroutine Return	RTN	—
Interrupt Request	IRQ	Interrupt Bit Pattern in 4 LS bits
Halt	HLT	—
Delay	DLY	Delay Time Value (resolution = 1 µs/LSB)
Wait Until Frame Timer = 0	WFT	—
Compare to Frame Timer	CFT	Delay Time Value (resolution = 100 µs/LSB)
Compare to Message Timer	CMT	Delay Time Value (resolution = 1 µs/LSB)
GP Flag Bits	FLG	Used to set, clear, or Toggle General Purpose flag bits
Load Time Tag Counter	LTT	Time Value
Load Frame Timer	LFT	Time Value (resolution = 100 µs/LSB)
Start Frame Timer	SFT	—
Push Time Tag Register	PTT	—
Push Block Status Word	PBS	—
Push Immediate Value	PSI	Immediate Value
Push Indirect	PSM	Memory Address
Wait for External Trigger	WTG	—
Execute and Flip	XQF	Message Control/Status Block Address

In the double buffered mode (FIGURE 6b), the RT automatically toggles between a pair of 32-word data buffers following each valid message received. This provides a means to ensure data sample consistency for applications in which the host processor's reading of received data blocks is asynchronous with the reception of 1553 messages (i.e., message reception does not cause an interrupt).

In the circular buffered mode (FIGURE 6c), the RT automatically increments the value of the lookup table pointer following each message transmitted or each valid message received. The RT may be programmed for individual circular buffers for specific transmit/receive subaddresses, and/or a global circular buffer for any subset of (or all) receive subaddresses. In addition, the Enhanced Mini-ACE RT may be programmed to issue interrupt requests when a specific circular buffer passes its 50% and/or 100% rollover points. The circular buffer mode may be used as a means to facilitate bulk data transfers, such as program downloads.

Other features of the Enhanced Mini-ACE RT include an RT address that may be specified by either hardware (via a card I/O connector) or by software; programmable command illegalization; and programmable busy by subaddress.

MONITOR MODE

The Enhanced Mini-ACE architecture includes a true message monitor. The monitor architecture provides programmable selection (filtering), based on the command word RT address/T-R bit/subaddress. As illustrated in FIGURE 7, the monitor generates a command stack and a separate data stack.

The monitor command stack includes a Block Status Word, which indicates the validity of the message; a time tag word; the received command word; and a pointer to the message's entry in the data stack, where the remainder of the message's words are stored. Similar to the operation for the RT circular buffer mode, there are 50% and 100% stack rollover interrupts provided for the command and data stacks.

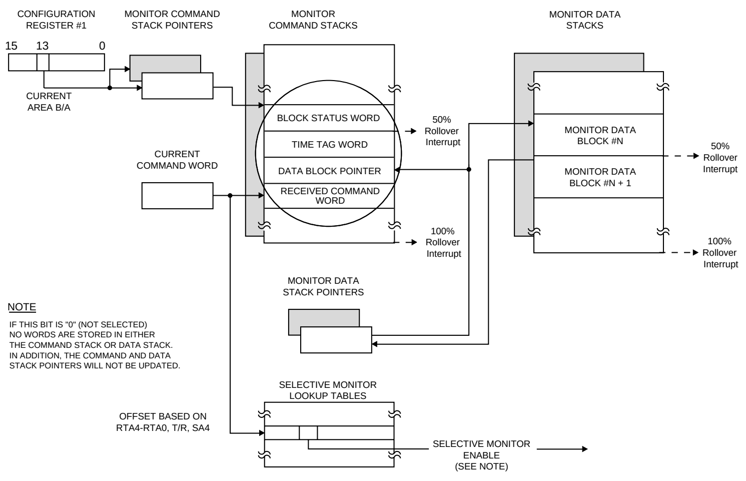
CONFIGURATION REGISTER #1
15 13 0
CURRENT AREA B/A
MONITOR COMMAND STACK POINTERS
MONITOR COMMAND STACKS
BLOCK STATUS WORD
TIME TAG WORD
DATA BLOCK POINTER
RECEIVED COMMAND WORD
50% Rollover Interrupt
100% Rollover Interrupt
CURRENT COMMAND WORD
MONITOR DATA STACKS
MONITOR DATA BLOCK #N
MONITOR DATA BLOCK #N + 1
MONITOR DATA STACK POINTERS
NOTE
IF THIS BIT IS "0" (NOT SELECTED) NO WORDS ARE STORED IN EITHER THE COMMAND STACK OR DATA STACK. IN ADDITION, THE COMMAND AND DATA STACK POINTERS WILL NOT BE UPDATED.
SELECTIVE MONITOR LOOKUP TABLES
OFFSET BASED ON RTA4-RTA0, T/R, SA4
SELECTIVE MONITOR ENABLE (SEE NOTE)

FIGURE 7. MONITOR MODE

INTERRUPT STATUS QUEUE

As illustrated in FIGURE 8, the Enhanced Mini-ACE RT and Monitor modes support an interrupt status queue. The interrupt status queue provides a mechanism for storing up to 32 of the most recent events that resulted in an interrupt condition. Each entry on the interrupt status queue consists of a vector word, which identifies the cause of the interrupt; along with a vector word, which points to the RT or Monitor command stack descriptor for the message that caused the interrupt. It is possible to perform filtering such that only valid messages and/or invalid messages result in entries on the interrupt status queue.

SELF-TEST

Another salient feature for the Enhanced Mini-ACE is the incorporation of a fully autonomous built-in self-test of protocol logic and shared RAM. These tests, which run automatically following power-up and may also be initiated by a host command, provide comprehensive tests of the terminal's internal protocol logic and RAM.

SOFTWARE LIBRARY, DRIVERS

The BU-69090 series Enhanced Mini-ACE software library functions and device drivers provide comprehensive support of the PMC and PC/104 1553 cards. The base library consists of a suite of function calls that serves to offload a great deal of low-level tasks from the application programmer. This includes register initialization, along with memory management software, and the means to implement an offline development environment.

As a means of supporting operation on multiple platforms, the BU-69090 library is written in ANSI C, and leverages component object modeling (COM). The use of ANSI C and component object modeling provides portability to different operating systems and card types. As a result, the library may be easily ported to run on platforms based on a variety of microprocessors, running under different operating systems or -- in some cases -- no operating system.

The BU-69090 software includes drivers for specific operating systems. These include VxWorks and Linux for the BU-65565 PMC card; along with DOS and VxWorks for the BU-65567/8 PC/104 cards. In addition, for DDC's PCI and PCMCIA laboratory cards, there are drivers for Windows 95/98, Windows NT, and Windows 2000.

The library is divided into two types of functions: external and internal. External functions are invoked by users, while internal functions are invoked by the library.

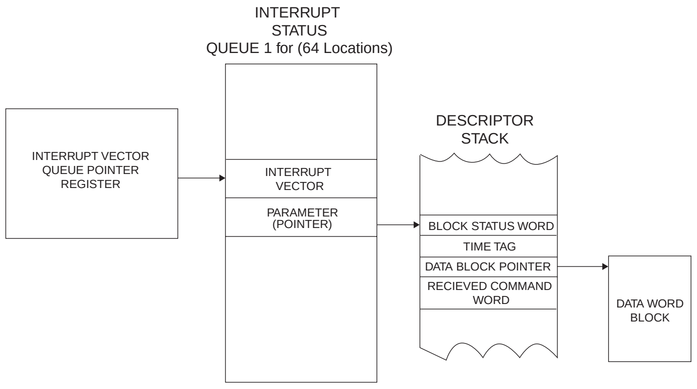

FIGURE 8. INTERRUPT STATUS QUEUE (FOR RT, MONITOR MODES)

The library allows the user to specify a unique device number, along with memory size, base register address, base memory address, and mode of operation for any Enhanced Mini-ACE on a given card. The library initialization function results in configuring the Enhanced Mini-ACEs to a specific state, depending on the mode of operation. For each mode, advanced architectural features are enabled as part of the initialization. Depending on the mode of operation that is initialized, the user may access specific data structures. For example, for BC mode, there are separate functions for accessing op codes, messages, data blocks, and frames. There are separate functions which may be invoked to release all resources for a particular device.

For all function calls, the library checks all parameters for validity, with invalid parameters resulting in error codes.

MEMORY MANAGEMENT SOFTWARE

The library operates under an open/access/close model. Under this model, areas of Enhanced Mini-ACE and host RAM are allocated and de-allocated by means of low-level routines. When an area of host RAM is closed, it is relinquished back to the operating system. While these low-level functions may be invoked directly by an application, in general their operation is transparent to the application programmer. The library's memory manager module performs autonomous allocation of shared memory for stacks, data blocks, and other structures. This provides a high degree of flexibility for sizing various data structures.

The library makes use of Global Structure Identifiers (GSIs). GSIs are used to identify unique structures within the Enhanced Mini-ACE library. A GSI is a user-assigned unique number that's usually specified by means of a '#define' directive. Once a structure is assigned a GSI, any reference to that particular GSI will modify that structure within the memory manager.

For example, for BC mode, these structures include op codes, messages, data blocks, and frames. The library includes an internal memory manager which allocates the next available location in memory to accommodate a particular structure. Memory management is implemented using a double linked list, which is the major structural component of the library. Individual nodes in the list are created and deleted by means of internal functions. A 'create node' or 'delete node' does not occur until the structure is needed for operation.

The memory management functions make use of handles consisting of starting addresses and sizes of memory blocks, along with status information delineating whether particular areas of shared RAM are unused, used, or protected. A handle is a pointer which is usually assigned by the operating system. A handle, which is global for the device driver, is assigned on a card type basis, and creates a link from the application program to the device driver. The driver deals with all inputs to and outputs from a card. In a system with multiple instances of the same type of card, each with multiple Enhanced Mini-ACEs, all cards/Enhanced Mini-ACEs work through one driver, via a single handle.

The various drivers are specific to a card type and a particular operating system. The interface between the library and the driver consists of a series of I/O control calls. These calls include Read_register, Write_register, Read_memory, Write_memory, Get_card_count, and Get_channel_count.

For each mode, there are functions to transfer data between shared RAM data blocks and buffers in host memory. In addition, there are functions to access consolidated data structures providing both message status information, as well as 1553 message words.

HOST BUFFERING

For all modes of operation, the Enhanced Mini-ACE runtime library allows the programmer to log all messages processed. With the use of the Host Buffering feature, all messages will be automatically transferred from the Enhanced Mini-ACE memory into a user-definable host memory segment. When polling in a real time operating system such as VxWorks, simple logging techniques such as polling may be used to capture all messages.

Unfortunately, in non-deterministic systems in which the user has little or no control of how long it will take to read new messages off the Enhanced Mini-ACE stack, messages may be lost.

In systems such as Microsoft Windows which do not have the luxury of real time processing, the runtime library allows for a Host Buffer to be installed. The Host Buffer (HBUF) is a circular memory structure resident on the host which contains the log of all messages. Messages are transferred to the HBUF by means of interrupts which occur at 50% and 100% rollover of RT circular buffers, or of the Monitor command and data stacks.

BC MODE SOFTWARE

The memory management for BC mode allocates RAM space for the BC instruction list, individual message control/status blocks, data blocks, and the general purpose queue.

The library provides comprehensive support of the Enhanced Mini-ACE's advanced bus controller capabilities. This includes function calls and macros invoking the BC instruction set.

The Enhanced Mini-ACE runtime library encapsulates all opcodes, data blocks, messages, and frames. Frame types include major, minor, and asynchronous. This allows the user to create the desired 1553 BC activity, without the overhead of memory management.

For BC mode, there are a number of linked list (LL) type constructs defined by the library. These include OpCode LL Items, Frame LL Items, Message LL Items, and Data Block LL Items. The library includes "create" and "delete" functions for these constructs.

The library also supports higher level bus controller functions. These implement higher level tasks such as minor and major frame timing control, conditional messaging, asynchronous message insertion, and interrupts after specific messages. There are also capabilities for interrupt-driven transfers of minor frame data and access to the general purpose queue.

RT MODE SOFTWARE

The library enables high-level operation for configuring and accessing the Enhanced Mini-ACE's RT. This includes routines for configuring the single, double, and circular subaddress buffering modes, and/or the global circular buffer mode. For each buffering mode, the library performs the necessary memory allocation and data block creation.

The library provides a mechanism for automatically reading and accessing the most recently received message. The library supports methods for synchronously and asynchronously accessing received message data using the single and double buffered methods respectively.

TABLE 2. A SAMPLING OF BC LIBRARY FUNCTIONS

FUNCTION	DESCRIPTION
aceBCDataBlkCreate(DevNum, nDataBlkID, wDataBlkSize,*pBuffer, wBufferSize)	This function allocates a data block to be used by any message. The data block may be 1 to 32 words long, single or double buffered.
aceBCOpCodeCreate (DevNum, nOpCodeID, wOpCodeType, wCondition, wParameter1, wParameter2, dwReserved)	This function creates an opcode/parameter word pair and appends it to the BC instruction list.
aceBCMsgCreateBCtoRT (DevNum, nMsgBlkID, nDataBlkID, wRT, wSA, wWC, MsgGapTime, dwMsgOptions)	This function creates the message control/status block for a BC-to-RT transfer message. There are separate functions for RT-to-BC transfers, RT-to-RT transfers, mode code messages, BC-to-RTs broadcast transfers, BC-to-RTs broadcast messages, and broadcast mode code messages.
AceBCFrameCreate (DevNum, nFrameBlkID, wFrameType, aOpCodeIDs, wOpCodeCount, wMnrFrmTime, wFlags)	This function creates a BC frame from an array of OpCode IDs. The frame may be either a minor or major frame.
AceBCStart (DevNum, nMjrFrmID, IMjrFrmCount, pMjrFrmNode, pMsgNode, pDataNode, pFrameNode)	This function initiates the BC to process a specified major frame.

TABLE 3. A SAMPLING OF RT LIBRARY FUNCTIONS

FUNCTION	DESCRIPTION
AceRTDataBlkCreate(DevNum, nDataBlkID, wDataBlkType,*pBuffer, wBufferSize)	This function allocates an RT data block.
AceRTDataBlkMapToSA (DevNum, nDataBlkID, wSA, wMsgType, wIrqOptions, wLegalizeSA)	This function maps a data block defined using aceRTDataBlkCreate with a specified transmit, receive, or broadcast subaddress. The function may also be used to legalize or illegalize the specified subaddress.
AceRTGetHBufMsgDecoded (DevNum, MSGSTRUCT *pMsg, *pdwMsgCount, *pdwMsgLostStk, *pdwMsgLostHBuf, wMsgLoc)	This function reads and decodes a message from the host buffer (assuming that one is present), and places the decoded message into the MSGSTRUCT parameter.
aceRTDataBlkCircBufInfo(DevNum, nDataBlkID, *pUserRWOffset, *pAceRWOffset)	This function returns information about a circular buffer, including the last read or written location performed by the user and the last location read or written by the Enhanced Mini-ACE RT.

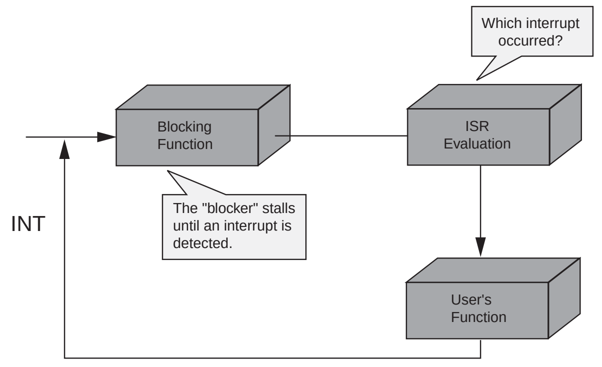
Which interrupt occurred?
Blocking Function
ISR Evaluation
INT
The "blocker" stalls until an interrupt is detected.
User's Function

FIGURE 9. INTERRUPT SERVICE “BLOCKING” FUNCTION

In addition, there is high-level support of subaddress illegalization and use of the busy bit (programmable by subaddress), enhanced mode code handling (interrupts and dedicated RAM locations for specific mode codes), along with functions allowing for accessing user-programmable status and BIT words.

The library includes constructs supporting bulk data transfers to or from an RT, using the Enhanced Mini-ACE circular buffer, and the 50% and 100% rollover interrupt features.

There is functionality for transferring one, multiple, or all RT messages to a host buffer. As discussed below, these functions store messages into consolidated data structures comprised of command and data words, and message status.

MESSAGE MONITOR SOFTWARE

The Enhanced Mini-ACE library’s message monitor architecture includes functions for specifying the monitor command and data stack sizes. This includes automatic allocation of RAM space for these stacks. The monitor library includes a function, (shown below), for programming of the monitor “select” or “filter” table. This allows the application to specify which 1553 commands, defined by specific combinations of RT addresses, T-R bit, and subaddress, will be stored by the monitor.

The monitor library includes high-level tools to decode monitored messages. Transfers from the monitor command and data stacks to a host buffer are interrupt-driven, using the 50% and 100% stack rollover interrupt features. Similar to the RT, the monitor software includes the capability to transfer message data words and status information to the host RAM in a consolidated data structure.

INTERRUPT HANDLING

Enhanced Mini-ACE interrupt handling involves the use of a blocking system (FIGURE 9). Interrupts are handled by a very high priority background thread, which is part of the library. The library sets up the thread. This routine normally stays in a “blocking” state, awaiting an interrupt request. At this “blocking” point, the thread waits in a dormant state for an interrupt to occur, or an “exit” condition to occur. An “exit” condition is either a BuClose, or is invoked by the operating system.

When the processor – and then the operating system – is signaled that an interrupt request has occurred, the driver reads card-specific registers to determine if a particular Enhanced Mini-ACE channel has requested interrupt service. Assuming that one of the Enhanced Mini-ACEs has issued an interrupt, the library checks to see which interrupt event(s) has occurred. If more than one interrupt event has occurred, these are processed sequentially, one at a time.

If the interrupt request was issued by a particular Enhanced Mini-ACE, then its “blocking” condition will be lifted. At this time, the library will then read the respective Enhanced Mini-ACE’s Interrupt Status Registers. Note that there is a separate thread for each Enhanced Mini-ACE. For each Interrupt Status Register bit that is set, the library interrupt service routine checks to see if the corresponding Interrupt Mask Register bit has been set. At this point, the library references a lookup table, which will then invoke one of several specific user routines associated with specific interrupt events. In responding to specific interrupt events, the user routines may then invoke other library functions.

After a particular interrupt event has been responded to, the thread will then perform a “manual” interrupt clear (if necessary). At that time, the thread will then revert to its “blocking” condition.

TABLE 4. A SAMPLING OF MONITOR LIBRARY FUNCTIONS

FUNCTION	DESCRIPTION
AceMTEnableRTFilter (DevNum, wRT, wTR, dwSAMask)	This function may be used to enable monitor selection for a specific subaddress or all subaddresses, for a specific RT address or all RT addresses.
AceMTStkToHBuf (DevNum)	This function copies all messages from the (previously active) stack to the host buffer. Once the messages have been moved to the host buffer, they can be processed by the application using either aceMTGetHBufMsgsRaw() or aceMTGetHBufMsgsDecoded().
AceMTGetHBufMsgDecoded (DevNum, MSGSTRUCT *pMsg, *pdwMsgCount, *pdwMsgLostStk, *pdwMsgLostHBuf, wMsgLoc)	This function reads either the last unread or most recently received decoded message from the host buffer.

FUNCTIONS TO TRANSFER DATA FROM ENHANCED MINI-ACE TO HOST

For each mode, there are functions to transfer data between shared RAM data blocks and buffers in host memory. In addition, there are functions to access consolidated data structures providing both message status information, as well as 1553 message words. The file Msgop.h, shown below, establishes the defined macros and structures for message decoding and handling.

DEFINED MACROS IN MSGOP.H

ACE_RX_CMD	*0x0000*	
ACE_TX_CMD	*0x0001*	
ACE_BROADCAST	*0X001F*	*/* Broadcast RT address (31)*/*
ACE_MODE_CODE1	*0x0000*	*/* Mode Code Subaddress (0)*/*
ACE_MODE_CODE2	*0x001F*	*/* Mode Code Subaddress (31)*/*
ACE_MSG_BCTORT	*0*	
ACE_MSG_RTTOBC	*1*	
ACE_MSG_RTTORT	*2*	
ACE_MSG_MODENODATA	*5*	
ACE_MSG_MODEDATARX	*6*	
ACE_MSG_MODEDATATX	*7*	
ACE_MSG_BRDCSTRTTORT	*10*	
ACE_MSG_BRDCSTMODENODATA	*13*	
ACE_MSG_BRDCSTMODEDATA	*14*	
ACE_MSG_INVALID	*15*	
ACE_MSGSIZE_RT	*36*	
ACE_MSGSIZE_MT	*40*	
ACE_MSGSIZE_BC	*42*	

In each mode, there are functions to decode raw messages. These functions take a 42-word buffer and decode the raw message into a decoded MSGSTRUCT. The raw message is the data read directly from the BC, RT, or monitor message. The MSGSTRUCT structure contains easily addressable elements of the message, as shown below.

STRUCTURES DEFINED IN MSGOP.H

```
Typedef struct MSGSTRUCT
{
    U16BIT wType;                   /* Contains the msg type (see above) */
    U16BIT wBlkSts;                 /* Contains the block status word */
    U16BIT wTimeTag;                /* Time tag of message */
    U16BIT wCmdWrd1;                /* First command word */
    U16BIT wCmdWrd2;                /* Second command word (RT to RT) */
    U16BIT wCmdWrd1Flg;             /* Is command word 1 valid? */
    U16BIT wCmdWrd2Flg;             /* Is command word 2 valid? */
    U16BIT wStsWrd1;                /* First status word */
    U16BIT wStsWrd2;                /* Second status word */
    U16BIT wStsWrd1Flg;             /* Is status word 1 valid? */
    U16BIT wStsWrd2Flg;             /* Is status word 2 valid? */
    U16BIT wWordCount;              /* Number of valid data words */
    U16BIT aDataWrds[32];           /* An array of data words */

    /* The following are only applicable in BC mode */
    U16BIT wBCCtrlWrd;              /* Contains the BC control word */
    U16BIT wBCGapTime;              /* Message gap time word */
    U16BIT wBCLoopBack1;            /* First looped back word */
    U16BIT wBCLoopBack2;            /* Second looped back word */
    U16BIT wBCLoopBack1Flg;         /* Is loop back 1 valid? */
    U16BIT wBCLoopBack2Flg;         /* Is loop back 2 valid? */

}MSGSTRUCT
```

VxWorks DRIVER

The Enhanced Mini-ACE software includes the BU-69090S2 VxWorks driver. There are separate versions of the driver for the BU-65565 PMC card and the BU-65567/8 PC/104 card. These drivers are designed to operate with version 5.2 of Wind River's VxWorks. The drivers for the BU-65565 PMC card were developed using a Motorola MVME 2700 card, which is based on a Power PC 750 processor. These drivers were developed using Wind River's Tornado II integrated development environment. The source code for the driver and library are provided to allow the driver to be tailored to any specific host board.

The VxWorks drivers for the BU-65567/8 PC/104 card were developed for an Intel Pentium environment.

The drivers make the required calls to the operating system necessary to acquire the correct address and interrupt resource information. These resources will be used by the driver during initialization of the card, and for read, write, and interrupt functions during normal operation. The ability to establish configuration in this manner enables a hands-off configuration of the card in any system.

OFFLINE DEVELOPMENT ENVIRONMENT

The software suite for the embedded cards includes support of an offline development environment (reference FIGURE 10). This allows code to be developed using a BU-65569I PCI card or BU-65553 PCMCIA card on an offline workstation such as a desktop PC, rather than on embedded system hardware.

The Enhanced Mini-ACE library includes a function that creates two files that download into the application program for the target embedded system. The first file is a binary file that contains an image of Enhanced Mini-ACE registers and memory, and the second file is a C++ header file that indicates all locations to structures/frames within memory. The binary image file is stored in the embedded system non-volatile memory. During initialization, it is then loaded into the Enhanced Mini-ACE shared RAM and registers.

The header file contains a readable ASCII representation of the entire mapping of the Enhanced Mini-ACE based on the operations entered into the program. The header file includes the location and size of message blocks and other data structures. In addition, it includes the source code for low-level functions to read and write Enhanced Mini-ACE Registers and RAM, as well as additional routines for particular functions; e.g., starting the bus controller.

The benefits of the use of image and header files include: (1) reduction in the size of embedded code; (2) reduced computational resources (CPU bandwidth cycles); and (3) the user has greater control over the development, validation, and documentation of flight critical and mission critical code.

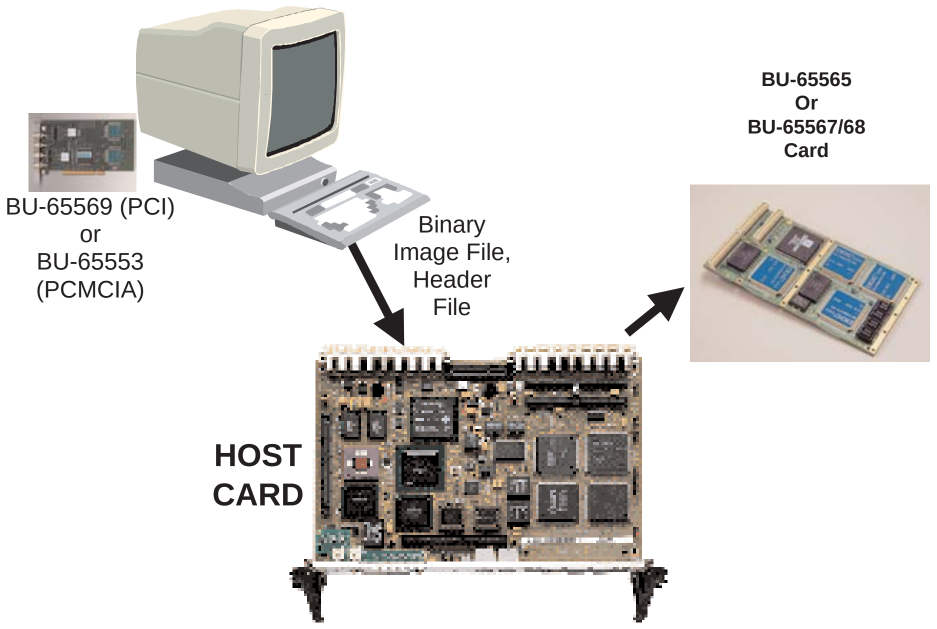

FIGURE 10. OFFLINE DEVELOPMENT ENVIRONMENT

105 Wilbur Place, Bohemia, New York, U.S.A. 11716-2426

For Technical Support - 1-800-DDC-5757 ext. 7771

Headquarters, N.Y., U.S.A. - Tel: (631) 567-5600, Fax: (631) 567-7358
Southeast, U.S.A. - Tel: (703) 450-7900, Fax: (703) 450-6610
West Coast, U.S.A. - Tel: (714) 895-9777, Fax: (714) 895-4988
United Kingdom - Tel: +44-(0)1635-811140, Fax: +44-(0)1635-32264
France - Tel: +33-(0)1-41-16-3424, Fax: +33-(0)1-41-16-3425
Germany - Tel: +49-(0)8141-349-087, Fax: +49-(0)8141-349-089
Japan - Tel: +81-(0)3-3814-7688, Fax: +81-(0)3-3814-7689
World Wide Web - http://www.ddc-web.com