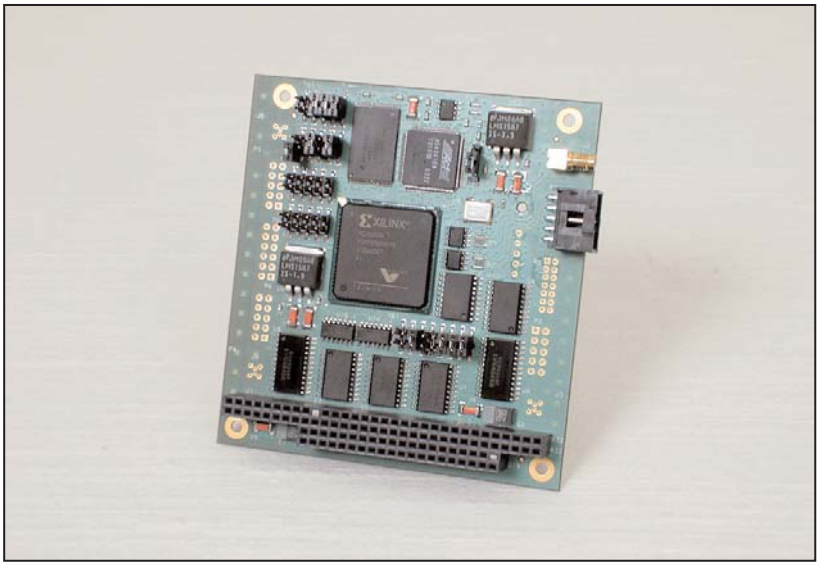
BU-6558XCX Series

ENHANCED BIT RATE-1553 BC/RT/MT PC/104 CARD

Make sure the next Card you purchase has...

DDC on Board®



DESCRIPTION

DDC's BU-6558XCX Series of PC/104 cards provide a complete COTS solution for interfacing between an embedded PC/104 bus and one to four Enhanced Bit Rate - 1553 Channel(s) with supporting CANBus interface(s), while also providing an optional MIL-STD-1553 dual redundant interface. Enhanced Bit Rate - 1553 (EBR-1553) is a 10Mbps data bus protocol based on the familiar MIL-STD-1553. The card's two addressing modes serve to provide compatibility with both Intel and Motorola processor platforms.

The BU-6558XCX includes one to four channel(s) of EBR-1553. The card's advanced Bus Controller (BC) architecture is software compatible with DDC's ACE software. This architecture provides a high degree of flexibility and autonomy. This includes methods to control message scheduling, the means to minimize host overhead for asynchronous message insertion, the ability to facilitate bulk data transfers and double buffering, and support for various message retry strategies. The BC architecture also provides flexibility for data logging and fault reporting.

The card's Remote Terminal (RT) architecture provides a great deal of flexibility. The choice of RT data buffering and interrupt options provides robust support for synchronous and asynchronous messaging, while ensuring data sample consistency and supporting bulk data transfers.

The BU-6558XCX Series cards also include one to four supporting CANBus interface(s) for MMSI EBR-1553 serial RT addressing and a 12-bit bi-directional discrete I/O port. The BU-65581CX adds an optional MIL-STD-1553 dual redundant interface.

FEATURES

- 16-bit Card per PC/104 Version 2.4
- One to Four EBR-1553 Channels
- MMSI Serial Addressing Capability with Supporting CANBus Interface(s)
- Optional Dual Redundant MIL-STD-1553 Interface (BU-65581CX)
- Convection Cooled with Extended Temperature Range Available
- Enhanced Bit Rate 1553 RT-only or BC/RT/MT Architecture
- 128K-word RAM
- Choice of Segmented or Flat Addressing Modes
- Highly Autonomous Bus Controller Architecture
- RT Buffering Options
- Selective Message Monitor
- Supports PC/104 Interrupts
- Software Library and Drivers for Windows® 2000/XP, VxWorks®, and DOS

Applications

- Embedded Systems
- Miniature Munitions/Store Interface (MMSI)
- Upgrade Path From MIL-STD-1553 to Higher Speed EBR-1553
- Bridging MIL-STD-1553 to MMSI EBR-1553
- Flight Recorders

DDC® Data Device Corporation

Data Device Corporation
105 Wilbur Place
Bohemia, New York 11716
631-567-5600 Fax: 631-567-7358
www.ddc-web.com

FOR MORE INFORMATION CONTACT:

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

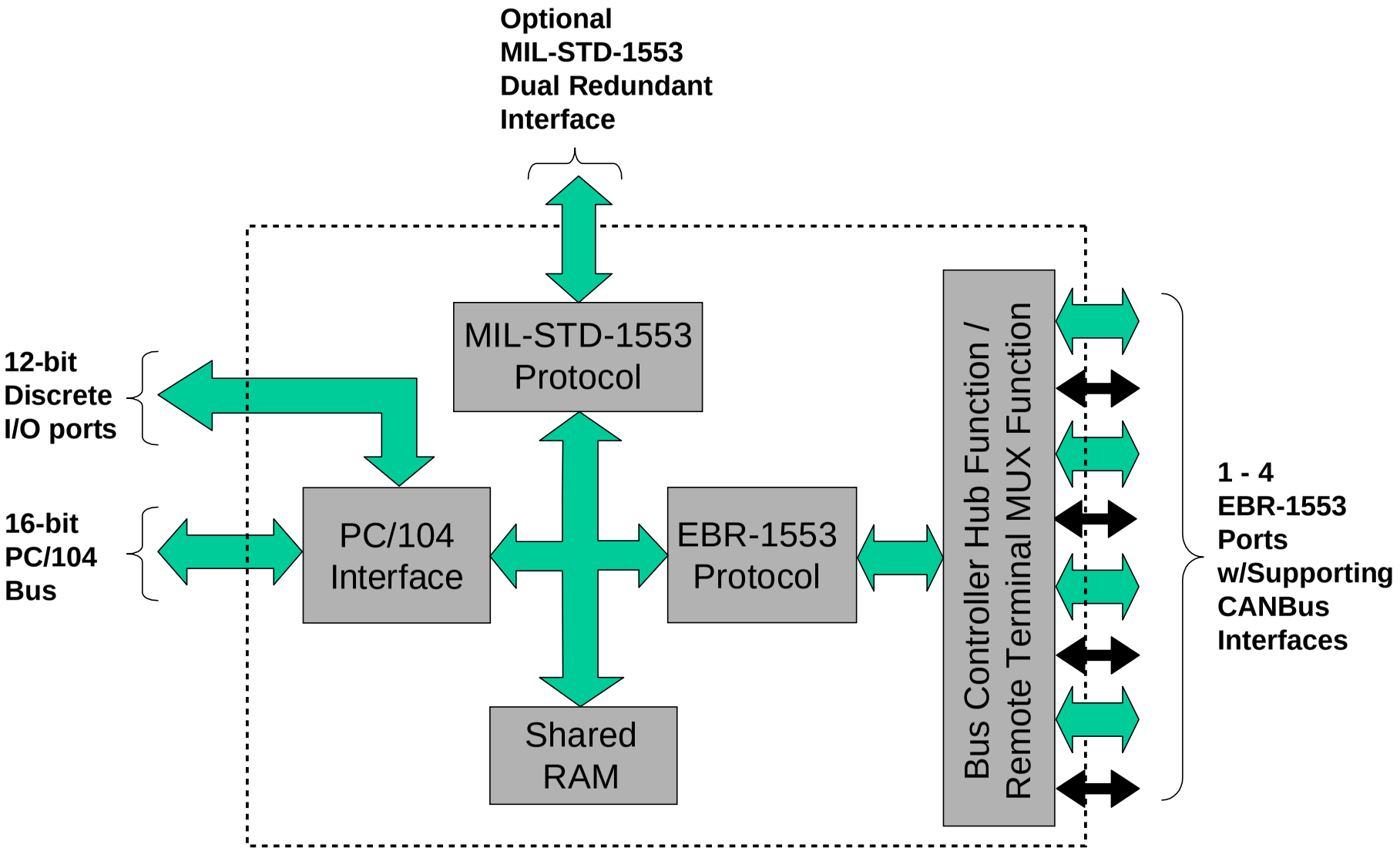


FIGURE 1. BU-6558XCX SERIES BLOCK DIAGRAM

TABLE 1. BU-6558XCX SPECIFICATIONS

PARAMETER	MIN	TYP	MAX	UNITS
ABSOLUTE MAXIMUM RATINGS				
Supply Voltage				
+5V	-0.3		6.0	V
I/O Input Voltage			5.5	V
I/O INPUT DRIVE LEVELS				
High Level Input Voltage	2			V
Low Level Input Voltage			0.8	V
I/O OUTPUT DRIVE LEVELS (@4mA)				
High Level Output Voltage	2.4			V
Low Level Output Voltage			0.4	V
POWER SUPPLY REQUIREMENTS				
Voltage/Tolerance				
+5V	4.75		5.25	V
Current Drain				
BU-65580C1				
Idle	200		380	mA
100% Duty Transmitter Cycle	325		600	mA
BU-65580C4				
Idle	275		550	mA
100% Duty Transmitter Cycle	400		700	mA
BU-65581C1				
Idle	TBD		TBD	mA
100% Duty Transmitter Cycle	TBD		TBD	mA
BU-65581C4				
Idle	TBD		TBD	mA
100% Duty Transmitter Cycle	TBD		TBD	mA
THERMAL				
Dissipation				
BU-65580C1				
Idle	1.000		1.990	W
100% Duty Transmitter Cycle	1.625		3.250	W
BU-65580C4				
Idle	1.375		2.900	W
100% Duty Transmitter Cycle	2.000		3.675	W
BU-65581C1				
Idle	TBD		TBD	W
100% Duty Transmitter Cycle	TBD		TBD	W
BU-65581C4				
Idle	TBD		TBD	W
100% Duty Transmitter Cycle	TBD		TBD	W
Board Operating Temperature*				
BU-6558XCX (Commercial)	0		55	°C
BU-6558XCX (Industrial)	-40		71	°C
Storage Temperature				
BU-6558XCX (Commercial)	-40		85	°C
BU-6558XCX (Industrial)	-62		85	°C
PHYSICAL CHARACTERISTICS				
Size	3.775 X 3.550 X 0.6 (98.9 X 90.2 X 15.2)			in. (mm)
Weight	3.5 (100)		4.1 (118)	oz. (g)

***Note:** Analysis has determined that an air flow of 1.5 m/sec is required to keep the junction temperature of the devices a minimum of 15°C below the maximum recommended by the component manufacturers.

INTRODUCTION

The BU-6558XCX Series of PC/104 cards provide one to four Enhanced Bit Rate - 1553 or EBR-1553 channel(s) on a convection-cooled card. The EBR-1553 Channel(s) are supported by CANBus interfaces to provide MMSI EBR-1553 serial RT addressing capability. EBR-1553 is a higher speed data bus protocol (10MHz) based on the familiar MIL-STD-1553 protocol.

The design of the BU-6558XCX Series leverages many capabilities of DDC's new ACE-Core. Features include a highly autonomous Bus Controller, a Remote Terminal providing a wide variety of buffering options, and selective message monitor. The EBR-1553 channel(s) contain 64K words of RAM.

Features of the board include:

- Channel independently programmed for BC, RT, Monitor, or RT/Monitor.
- Convection cooled.
- 128K words SRAM.

The BU-6558XCX Series is supported by free software, including a "C" runtime library, VxWorks and DOS driver. This software supports most of the ACE-Core's advanced architectural features.

One of the new salient features of the ACE-Core based product is its new Bus Controller architecture. The Enhanced Bus Controller's highly autonomous message sequence control engine provides a means for offloading the host processor for implementing multi-frame message scheduling, message retry, data double buffering, and asynchronous message insertion. In addition, the Enhanced BC mode includes eight general-purpose flag bits, a general-purpose queue, and user definable interrupts for the purpose of performing messaging to the host.

The Remote Terminal offers choice of single, double, and circular buffering for individual subaddresses or a global circular buffering option for multiple (or all) receive subaddresses, 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.

TRANSCEIVERS

The BU-6558XCX Series uses high-speed, low-power RS-485 differential bus transceivers as called out in the MMSI/EBR-1553 specifications.

TABLE 2. DISCRETE I/O REGISTER 0 (OFFSET ADDRESS 01A6) REGISTER PROVIDES EIGHT BITS OF DISCRETE INPUT/OUTPUT (BIT 14 = "0")			
BIT	**FUNCTION**		
BIT [15]	(R/W) DIO Direction ("0" = inputs, "1" = outputs for BITS 7-0) DIO Direction can be either a "0" or "1" when in External Hub and DIO mode		
BIT [14]	(R/W) DIO Function ("0" = DIO, "1" = External HUB and DIO)		
BIT [13 : 8]	Spare (R/W)		
—	If BIT (14) = "0" (DIO)	If BIT(14) = "1" (External Hub and DIO)	
BIT [7]	DIO_7 (read only if discrete input; R/W if discrete output)	A4 (MSB)	BITS (7:3, A4 thru A0)are the HUB Address bits to allow addressing of up to 31 ports. Port addresses 0 through 3 are used on the PC/104 board and Port addresses 4 through 30 can be used externally. Port 31 is reserved for broadcast.
BIT [6]	DIO_6 (read only if discrete input; R/W if discrete output)	A3	
BIT [5]	DIO_5 (read only if discrete input; R/W if discrete output)	A2	
BIT [4]	DIO_4 (read only if discrete input; R/W if discrete output)	A1	
BIT [3]	DIO_3 (read only if discrete input; R/W if discrete output)	A0 (LSB)	
BIT [2]	DIO_2 (read only if discrete input; R/W if discrete output)	EXT TX DATA (Serial data output to be used for external hub expansion)	
BIT [1]	DIO_1 (read only if discrete input; R/W if discrete output)	DIO Output	
BIT [0]	DIO_0 (read only if discrete input; R/W if discrete output)	DIO Output	

TABLE 3. DISCRETE I/O REGISTER 1 (OFFSET ADDRESS 0X01A8) REGISTER PROVIDES FOUR BITS OF DISCRETE INPUT/OUTPUT		
BIT	**FUNCTION**	
BIT [15]	(R/W) DIO Direction ("0" = inputs, "1" = outputs for BIT 3:0) DIO Direction can be either a "0" or "1" when in External Hub and DIO mode	
BIT [14 : 4]	Spare (R/W)	
—	If BIT (14) of Discrete I/O Register 0 (offset address 01A6) = "1" (DIO)	If BIT (14) of Discrete I/O Register 0 (offset address 01A6) = "0" (External Hub and DIO)
BIT [3]	DIO_11 (read only if discrete input; R/W if discrete output)	DIO Input
BIT [2]	DIO_10 (read only if discrete input; R/W if discrete output)	DIO Input
BIT [1]	DIO_9 (read only if discrete input; RW if discrete output)	DIO Input
BIT [0]	DIO_8 (read only if discrete input; RW if discrete output)	EXT RX DATA (Serial data input used for Hub expansion)

INTERRUPTS

The ACE-Core may issue interrupt requests over the PC/104 bus. The PC/104 interrupt level is user-programmable from among levels 3, 4, 5, 7, 10, 11, 12, 14, or 15. The interrupt level is user-selectable by means of the base memory address I/O register. The interrupt outputs are open collector type signals, which provide 0.5µs negative-going pulses.

SOFTWARE

The DOS and Windows 2000/XP support is provided with the legacy BUS-69080 ACE software runtime library. The BU-6558XCX VxWorks drivers and libraries are based on the Enhanced Mini-ACE BU-69090 software runtime library. The Enhanced Mini-Ace Runtime Library provides comprehensive support of the BU-6558XCX Series of PC/104 cards. This driver and RTL comprises a suite of function calls that serve to offload a great deal of the low level tasks from the application programmer.

MEMORY ALLOCATION

For each mode of operation - BC, RT, and Monitor - the library and driver operate under an open/access/close model, in which areas of RAM are autonomously allocated and de-allocated by means of low level routines. While these 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

TABLE 4. EBR/MIL-STD-1553 REGISTERS					
ADDRESS					REGISTER
A4	A3	A2	A1	A0	DESCRIPTION / ACCESSIBILITY
0	0	0	0	0	Interrupt Mask Register #1 (RD/WR)
0	0	0	0	1	Configuration Register #1 (RD/WR)
0	0	0	1	0	Configuration Register #2 (RD/WR)
0	0	0	1	1	Start/Reset Register (WR)
0	0	0	1	1	Non-Enhanced BC or RT Command Stack Pointer/Enhanced BC Instruction List Pointer Register (RD)
0	0	1	0	0	BC Control Word/RT Subaddress Control Word Register (RD/WR)
0	0	1	0	1	Time Tag Register (RD/WR)
0	0	1	1	0	Interrupt Status Register #1 (RD)
0	0	1	1	1	Configuration Register #3 (RD/WR)
0	1	0	0	0	Configuration Register #4 (RD/WR)
0	1	0	0	1	Configuration Register #5 (RD/WR)
0	1	0	1	0	RT/Monitor Data Stack Address Register (RD/WR)
0	1	0	1	1	BC Frame Time Remaining Register (RD)
0	1	1	0	0	BC Time Remaining to Next Message Register (RD)
0	1	1	0	1	Non-Enhanced BC Frame Time/Enhanced BC Initial Instruction Pointer/RT Last Command/MT Trigger Word Register (RD/WR)
0	1	1	1	0	RT Status Word Register
0	1	1	1	1	RT BIT Word Register (RD)
1	0	0	0	0	Test Mode Register 0
1	0	0	0	1	Test Mode Register 1
1	0	0	1	0	Test Mode Register 2
1	0	0	1	1	Test Mode Register 3
1	0	1	0	0	Test Mode Register 4
1	0	1	0	1	Test Mode Register 5
1	0	1	1	0	Test Mode Register 6
1	0	1	1	1	Test Mode Register 7
1	1	0	0	0	Configuration Register #6 (RD/WR)
1	1	0	0	1	Configuration Register #7 (RD/WR)
1	1	0	1	0	Reserved
1	1	0	1	1	BC Condition Code Register (RD)
1	1	0	1	1	BC General Purpose Flag Register (WR)
1	1	1	0	0	BIT Test Status Register (RD)
1	1	1	0	1	Interrupt Mask Register #2 (RD/WR)
1	1	1	1	0	Interrupt Status Register #2 (RD)
1	1	1	1	1	BC General Purpose Queue Pointer/RT-MT Interrupt Status Queue Pointer Register (RD/WR)

memory for stacks and data blocks. This provides a high degree of flexibility for sizing various data structures.

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. 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.

MODE-SPECIFIC SOFTWARE

In BC mode, there is comprehensive support of the enhanced Bus Controller capabilities, allowing the user to leverage function calls and macros invoking the BC instruction set. Some of the functions support higher level tasks such as minor and major frame timing control and asynchronous message insertion. For the enhanced BC, the software also supports the offline development compilation of BC message scenarios on (for example) a desktop PC. The binary images created from these may then be downloaded to the target processor environment.

In BC mode, the Enhanced Mini-ACE runtime library encapsulates all op codes, data blocks, messages, and frames (major, minor, and asynchronous). This allows the user to create the desired 1553 BC activity, without the overhead of memory management schemes. The library includes a function that creates two files, which allow the user to view op code sequences generated by the library. The first file is a binary file, which contains an image of ACE memory, and second is a 'C' file that shows all locations to structures/frames within memory.

In RT mode, there is high-level operation for configuring and utilizing the ACE-Core's single-buffer, double-buffer, and circular buffered subaddress memory management schemes. This includes methods for accessing synchronously and asynchronously received message data. There is also a mechanism provided for automatically reading and accessing the most recently received message data. In addition, there is high-level support of subaddress illegalization and use of the busy bit, enhanced mode code handling, along with functions allowing for accessing user programmable status and BIT words.

In the message monitor mode, functions are provided for programming the command and data stack sizes, programming of the monitor “filter” table (which addresses/T-R/subaddresses to monitor), along with high-level tools that decode monitored messages, and transfer status information and message words to host RAM in a consolidated stack data structure.

IMAGE FILES

As a means of reducing both the code size and the level of computational resources for embedded system software, the BU-69080C1 enables the library routines used to initialize the ACE-Core to be processed in an offline (non-embedded) development environment. By limiting the library's use to the non-flight environment, the user has greater control over the development and validation of the code for the embedded system.

The output of this process is a pure binary image file for the ACE-Core registers and shared RAM, along with the header file information. The header file information provides data about the location and size of various data structures, along with the source code for the four low-level functions to read and write the ACE-Core registers and RAM.

The binary image may then be downloaded into the source of the embedded host program. As a result, to initialize the 1553 channel, all that the executable program needs to do is write the image file to the ACE-Core registers and shared RAM. The header file information may then be used as a mechanism for determining the location and size of message blocks and other data structures.

I/O-MAPPED REGISTERS, MEMORY-MAPPED REGISTERS, AND MEMORY MAPPING

The BU-6558XCX Series contains a number of registers. Two of these are I/O-mapped, while the rest are memory mapped. The I/O base address is jumper selectable. The card's base memory address is programmable by means of the EBR/MIL-STD-1553 Memory Base Address/Interrupt Level Register. This register also is used for selecting the card’s interrupt level.

The base address of the card’s memory mapped registers is programmable by means of the Register Base Address Register. The memory mapped registers include all of the registers in the individual ACE-Cores. Note that the ACE-Core register space is 64 registers. There are also several additional memory mapped registers for the card including: the Card ID Register, Channel/Segment Select Register, Interrupt Status Register, and two Discrete I/O registers.

The EBR/MIL-STD-1553 Memory Base Address/Interrupt Level Register contains a MEMORY ENABLE bit, while the Register Base Address Register contains a REGISTER ENABLE bit. These bits, which default to logic “0” following Power On, must be set to logic “1” by the PC/104 host in order to enable host access to the card’s memory and registers. Note that while either or both of these bits are still logic “0”, the card will continue to operate on the 1553 bus (if previously configured); however the card’s registers and/or memory will not be host accessible.

The Card ID Register identifies whether the paged or flat mode addressing is enabled. The Interrupt Status Register signals the host which resource is requesting service.

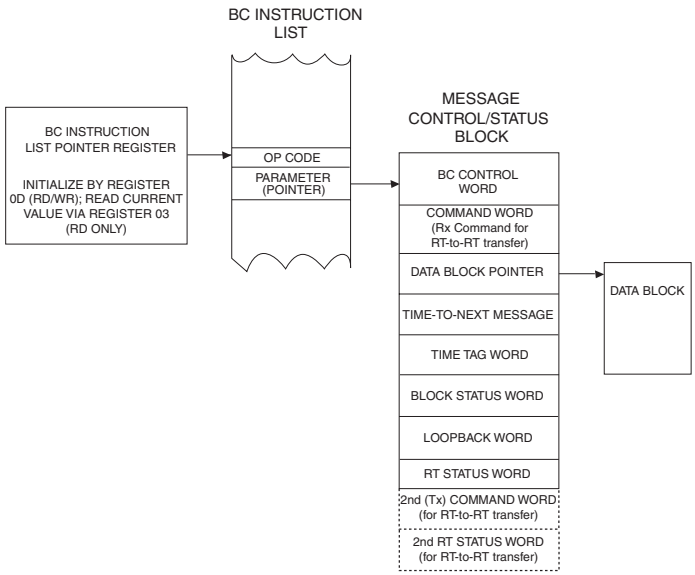


FIGURE 2. BC MESSAGE SEQUENCE CONTROL

GENERAL PURPOSE I/O PORTS

The BU-6558XCX Series includes two discrete I/O ports. The ports consist of an 8-bit port and a 4-bit port. These ports are memory mapped to the PC/104 host via the two Discrete I/O Registers. Bit 15 of these registers is used to specify the direction of each I/O Port (0 = input, 1 = output). Bit 15 is read/writable. The lower bits control or enable reading of the respective I/O signal logic sense. Output bits are host read/writable. Lower bits programmed as inputs are read-only. Table 2 and Table 3 illustrate the bit mapping for the two Discrete I/O Registers.

BUS CONTROLLER (BC) ARCHITECTURE

The EBR-1553 Bus Controller is either a single interface or a four port EBR-1553 Hub. The four port Bus Controller Hub function allows the EBR-1553 Bus Controller to communicate with up to four Remote Terminals.

The BC functionality for both the EBR-1553 channel and the Optional MIL-STD-1553 Channel includes two separate architectures: (1) the older, non-Enhanced mode, which provides complete compatibility with the previous ACE and Mini-ACE (Plus) generation products; and (2) the newer, Enhanced BC mode. The enhanced BC mode offers several new powerful architectural features. These include the incorporation of a highly autonomous BC message sequence control engine, which greatly serves to offload the host CPU.

The Enhanced BC's message sequence control engine provides a high degree of flexibility for implementing major and minor frame scheduling; capabilities for inserting asynchronous messages in the middle of a frame; to separate 1553 message data from control/status data for the purpose of implementing double buffering and performing bulk data transfers; for implementing message retry schemes, and for reporting various conditions to the host processor by means of four user defined interrupts and a general purpose queue.

In both the non-Enhanced and Enhanced BC modes, the ACE-Core BC implements all EBR-1553 message formats. Message format is programmable on a message-by-message basis by means of the BC Control Word and the T/R bit of the Command Word for the respective message. The BC control Word allows 1553 message format, bus channel, self-test, and Status Word masking to be specified on an individual message basis. In addition, automatic retries and/or interrupt requests may be enabled or disabled for individual messages. The BC performs all error checking required by EBR-1553. This includes validation of response time, sync type and sync encoding, Manchester II encoding, parity, bit count, word count, and Status Word RT Address field. The BC response time is programmable.

In its non-Enhanced mode, the ACE-Core may be programmed to process BC frames of up to 512 messages with no processor intervention. In the Enhanced BC mode, there is no explicit limit to the number of messages that may be processed in a frame. In both modes, it is possible to program for either single frame or frame auto-repeat operation. In the auto-repeat mode, the frame repetition rate may be controlled either internally, using a programmable BC frame timer, or from an external trigger output.

ENHANCED BC MODE: MESSAGE SEQUENCE CONTROL

One of the major architectural features of the Enhanced Mini-ACE series is its advanced capability for BC message sequence control. The ACE-Core supports highly autonomous BC operation, which greatly offloads the operation of the host processor.

The operation of the ACE-Core's message sequence control engine is illustrated in FIGURE 2. The BC message sequence control involves an instruction list pointer register, an instruction list, which contains multiple 2 word entries, a message control/status stack, which contains multiple 8 word or 10 word descriptors, and data blocks for individual messages.

The value of the instruction list pointer register is initialized by the host processor (via Register 0x0D), and is incremented by the BC message sequence processor (host readable via Register 0x03). During operation, the message sequence control processor fetches the operation referenced by the instruction list pointer register from the instruction list.

Note that the pointer parameter referencing the first word of a message's control/status block (the BC Control Word) must contain an address value that is modulo 8.

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Odd Parity	OpCode Field					0	1	0	1	0	Condition Code Field				

FIGURE 3. BC OP CODE FORMAT

OP CODES

The instruction list pointer references a pair of words in the BC instruction list: an op code word, followed by a parameter word. The format of the op code word, which is illustrated in FIGURE 3, includes a 5-bit op code field and a 5-bit condition code field. The op code identifies the instruction to be executed by the BC message's sequence control engine.

Most of the operations are conditional, with the execution dependent on the contents of the condition code field. Bits[3:0] of the condition code field identifies the particular condition. Bit 4 of the condition code field identifies the logic sense ("1" or "0") of the selected condition code on which the conditional execution is dependent. TABLE 5 lists all op codes along with their respective mnemonic, code value, parameter, and description. TABLE 6 defines all the condition codes.

Eight of the conditional codes (0x8 through 0xF) are set or cleared as the result of the most recent message. The other eight are defined as "General Purpose" condition codes GP0-GP7. There are three mechanisms for programming the values of the General Purpose Condition Code bits: (1) They may be set, cleared, or toggled by the host processor by means of the BC GENERAL PURPOSE FLAG REGISTER; (2) they may be set, cleared, or toggled by the BC message sequence control processor by means of the GP Flag Bits (FLG) instruction; and (3) GPO and GP1 only (no others) may be set or cleared by means of the BC message sequence control processor's Compare Frame Timer (CFT) or Compare Message Timer (CMT) instructions.

The host processor also has read-only access to the BC condition codes by means of the BC CONDITION CODE REGISTER.

Note that four (4) of the instructions are unconditional. These are the Compare to Frame Timer (CFT), Compare to Message Timer (CMT), GP Flag Bits (FLG), and Execute and Flip (XQF). For these instructions, the Condition Code field is "don't care". These instructions are always executed regardless of the result of the condition code test. All other instructions are conditional. They will only be executed if the condition code specified by the Condition Code field in the op code word tests true. If the condition code field tests false, the instruction list pointer will skip down to the next instruction.

As shown in TABLE 5, many of the operations include a single word parameter. For an XEQ (execute message) operation, the parameter is a pointer to start of the message's control/status block. For other operations, the parameter may be an address, a time value, an input pattern, a mechanism to set or clear General Purpose Flag bits, or an immediate value. For several op codes the parameter is a "don't care" and is not used.

As described above, some of the op codes will cause the message sequence control engine to execute messages. In this case, the parameter references the first word of a message control/status block. All message status/control blocks are eight words long; a block control word, time-to-next-message parameter, data block pointer, command word, status word, loopback word, block status word, and a time tag word.

The third word in the message control/status block is a pointer that references the first word of the message's data word block. Note that the data word block stores only data words, which are to be either transmitted or received by the BC. By segregating data words from command words, status words, and other control and "housekeeping" functions, this architecture enables the use of convenient, usable data structures, such as circular buffers and double buffers.

Other operations support program flow control; i.e., jump and call capability. The call capability includes maintenance of a call stack that supports a maximum of four (4) entries, and there is also a return instruction. In the case of a call stack overrun or underrun, the BC will issue an CALL STACK POINTER REGISTER ERROR interrupt, if enabled.

Other op codes may be used to delay for a specified time, start a new BC frame, wait for an external trigger to start a new frame, do comparisons based on frame time and time-to-next message, load the time tag or frame time registers, halt, and issue host interrupts. In the case of host interrupts, the message sequence control engine passes a 4-bit user-defined interrupt vector to the host by means of the Interrupt Status Register.

The purpose of the FLG instruction is to enable the message sequence controller to set, clear, or toggle the value(s) of any or all of the eight general purpose condition flags.

The op code parity bit encompasses all sixteen bits of the op code word. This bit must be programmed for odd parity. If the message sequence control processor fetches an undefined op code word, an op code word with even parity, or bits[9:5] of an op code word do not have a binary pattern of 01010, the message sequence control processor will immediately halt the BC's operation. In addition, if enabled, a BC TRAP OP CODE interrupt will be issued. Also, if enabled, a parity error will result in an OP CODE PARITY ERROR interrupt.

TABLE 5. BC OPERATIONS FOR MESSAGE SEQUENCE CONTROL					
INSTRUCTION	**MNEMONIC**	**OP CODE (HEX)**	**PARAMETER**	**CONDITIONAL OR UNCONDITIONAL**	**DESCRIPTION**
Execute Message	XEQ	0x0001	Message Cntrl /Status Block Address	Conditional (See NOTE)	Execute the message at the specified Message Control/Status Block Address if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Jump	JMP	0x0002	Instruction List Address	Conditional	Jump to the Op Code specified in the Instruction List if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Subroutine Call	CAL	0x0003	Instruction List Address	Conditional	Jump to the Op Code specified by the Instruction List Address and push the Address of the Next Op Code on the Call Stack if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list. Note that the maximum depth of the subroutine call stack is four.
Subroutine Return	RTN	0x0004	Not Used (don't care)	Conditional	Return to the Op Code popped off the Cal Stack if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Interrupt Request	IRQ	0x0006	Interrupt Bit Pattern in 4 LS bits	Conditional	Generate an interrupt if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list. The passed parameter (IRQ Bit Pattern) specifies which of the ENHANCED BC IRQ bit(s) (bits 5-2) will be set in Interrupt Status Register #2. Only the four LSBs of the passed parameter are used. A parameter where the four LSBs are logic "0" will not generate an interrupt.
Halt	HLT	0x0007	Not Used (don't care)	Conditional	Stop execution of the Message Sequence Control Program until a new BC Start is issued by the host if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Delay	DLY	0x0008	Delay Time Value (resolution = 1 µs/LSB)	Conditional	Delay the time specified by the Time parameter before executing the next Op Code if the condition flag tests TRUE, otherwise continue execution at the next Op Code without delay. The delay generated will use the Time to Next Message Timer.
Wait Until Frame Timer = 0	WFT	0x0009	Not Used (don't care)	Conditional	Wait until Frame Time counter is equal to zero before continuing execution of Message Sequence Control Program if the condition flag tests TRUE, otherwise continue execution at the next Op Code without delay.
Compare to Frame Timer	CFT	0x000A	Delay Time Value (resolution = 100 µs/LSB)	Unconditional	Compare Time Value to Frame Time Counter. The LT/GP0 and EQ/GP1 flag bits are set or cleared based on the results of the compare. If the value of the CFT's parameter is less than the value of the frame time counter, then the LT/GP0 and NE/$\overline{GP1}$ flags will be set, while the GT-EQ/$\overline{GP0}$ and EQ/GP1 flags will be cleared. If the value of the CFT's parameter is equal to the value of the frame time counter, then the GT-EQ/$\overline{GP0}$ and EQ/GP1 flags will be set, while the LT/GP0 and NE/$\overline{GP1}$ flags will be cleared. If the value of the CFT's parameter is greater than the current value of the frame time counter, then the GT-EQ/$\overline{GP0}$ and NE/$\overline{GP1}$ flags will be set, while the LT/GP0 and EQ/GP1 flags will be cleared.
Compare to Message Timer	CMT	0x000B	Delay Time Value (resolution = 1 µs/LSB)	Unconditional	Compare Time Value to Message Time Counter. The LT/GP0 and EQ/GP1 flag bits are set or cleared based on the results of the compare. If the value of the CMT's parameter is less than the value of the message time counter, then the LT/GP0 and NE/$\overline{GP1}$ flags will be set, while the GT-EQ/$\overline{GP0}$ and EQ/GP1 flags will be cleared. If the value of the CMT's parameter is equal to the value of the message time counter, then the GT-EQ/$\overline{GP0}$ and EQ/GP1 flags will be set, while the LT/GP0 and NE/$\overline{GP1}$ flags will be cleared. If the value of the CMT's parameter is greater than the current value of the message time counter, then the GT-EQ/$\overline{GP0}$ and NE/$\overline{GP1}$ flags will be set, while the LT/GP0 and EQ/GP1 flags will be cleared.

Note: While the XEQ (Execute Message) instruction is conditional, not all condition codes may be used to enable its use. The ALWAYS and NEVER condition codes may be used. The eight general purpose flag bits, GP0 through GP7, may also be used. However, if GP0 through GP7 are used, it is imperative that the host processor not modify the value of the specific general purpose flag bit that enabled a particular message while that message is being processed. Similarly, the LT, GT-EQ, EQ, and NE flags, which the BC only updates by means of the CFT and CMT instructions, may also be used. However, these two flags are dual use. Therefore, if these are used, it is imperative that the host processor not modify the value of the specific flag (GP0 or GP1) that enabled a particular message while that message is being processed. The NORESP, FMT ERR, GD BLK XFER, MASKED STATUS SET, BAD MESSAGE, RETRY0, and RETRY1 condition codes are not available for use with the XEQ instruction and should not be used to enable its execution.

TABLE 5. BC OPERATIONS FOR MESSAGE SEQUENCE CONTROL (CONT)

INSTRUCTION	MNEMONIC	OP CODE (HEX)	PARAMETER	CONDITIONAL OR UNCONDITIONAL	DESCRIPTION
GP Flag Bits	FLG	0x000C	Used to set, clear, or Toggle GP (General Purpose) flag bits (see description)	Unconditional	Used to set, toggle, or clear any or all of the eight general purpose flags. The table below illustrates the use of the GP Flag Bits instruction for the case of GP0 (General Purpose Flag 0). Bits 1 and 9 of the parameter byte affect flag GP1, bits 2 and 10 affect GP2, etc., according to the following rules: **BIT 8** / **BIT 0** / **EFFECT ON GPO** 0 / 0 / No Change 0 / 1 / Set Flag 1 / 0 / Clear Flag 1 / 1 / Toggle Flag
Load Time Tag Counter	LTT	0x000D	Time Value. Resolution (µs/LSB) is defined by bits 9, 8, and 7 of Configuration Register #2	Conditional	Load Time Tag Counter with Time Value if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Load Frame Timer	LFT	0x000E	Time Value (resolution = 100 µs/LSB)	Conditional	Load Frame Timer Register with the Time Value parameter if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Start Frame Timer	SFT	0x000F	Not Used (don't care)	Conditional	Start Frame Time Counter with Time Value in Time Frame register if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Push Time Tag Register	PTT	0x0010	Not Used (don't care)	Conditional	Push the value of the Time Tag Register on the General Purpose Queue if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Push Block Status Word	PBS	0x0011	Not Used (don't care)	Conditional	Push the Block Status Word for the most recent message on the General Purpose Queue if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Push Immediate Value	PSI	0x0012	Immediate Value	Conditional	Push Immediate data on the General Purpose Queue if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Push Indirect	PSM	0x0013	Memory Address	Conditional	Push the data stored at the specified memory location on the General Purpose Queue if the condition flag tests TRUE, otherwise continue execution at the next Op Code in the instruction list.
Wait for External Trigger	WTG	0x0014	Not Used (don't care)	Conditional	Wait until a logic "0"-to-logic "1" transition on the EXT_TRIG input signal before proceeding to the next Op Code in the instruction list if the condition flag tests TRUE, otherwise continue execution at the next Op Code without delay.
Execute and Flip	XQF	0x0015	Message Control/Status Block Address	Unconditional	Execute (unconditionally) the message for the Message Control/Status Block Address. Following the processing of this message, if the condition flag tests TRUE, then flip bit 4 in the Message Control/Status Block Address, and store the new Message Block Address as the updated value of the parameter following the XQF instruction code. As a result, the next time that this line in the instruction list is executed, the Message Control/Status Block at the updated address (old address XOR 0010h), rather than the old address, will be processed.

TABLE 6. BC CONDITION CODES

BIT CODE	NAME (BIT 4=0)	INVERSE (BIT 4=1)	FUNCTIONAL DESCRIPTION
0000	LT/GP0	GT-EQ/$\overline{\text{GP0}}$	Less than or GP0 flag. This bit is set or cleared based on the results of the compare. If the value of the CMT's parameter is less than the value of the message time counter, then the LT/GP0 and NE/$\overline{\text{GP1}}$ flags will be set, while the GT-EQ/$\overline{\text{GP0}}$ and EQ/GP1 flags will be cleared. If the value of the CMT's parameter is equal to the value of the message time counter, then the GT-EQ/$\overline{\text{GP0}}$ and EQ/GP1 flags will be set, while the LT/GP0 and NE/$\overline{\text{GP1}}$ flags will be cleared. If the value of the CMT's parameter is greater than the current value of the message time counter, then the GT-EQ/$\overline{\text{GP0}}$ and NE/$\overline{\text{GP1}}$ flags will be set , while the LT/GP0 and EQ/GP1 flags will be cleared. Also, General Purpose Flag 1 may also be set or cleared by a FLG operation.
0001	EQ/GP1	NE/$\overline{\text{GP1}}$	Equal Flag. This bit is set or cleared after CFT or CMT operation. If the value of the CMT's parameter is equal to the value of the message time counter, then the EQ/GP1 flag will be set and the NE/$\overline{\text{GP1}}$ bit will be cleared. If the value of the CMT's parameter is not equal to the value of the message time counter, then the NE/$\overline{\text{GP1}}$ flag will be set and the EQ/GP1 bit will be cleared. Also, General Purpose Flag 1 may also be set or cleared by a FLG operation.
0002	GP2	$\overline{\text{GP2}}$	General Purpose Flags set or cleared by FLG operation or by host processor. The host processor can set, clear, or toggle these flags in the same way as the FLG instruction by means of the BC GENERAL PURPOSE FLAG REGISTER.
0003	GP3	$\overline{\text{GP3}}$	
0004	GP4	$\overline{\text{GP4}}$	
0005	GP5	$\overline{\text{GP5}}$	
0006	GP6	$\overline{\text{GP6}}$	
0007	GP7	$\overline{\text{GP7}}$	
0008	NORESP	$\overline{\text{RESP}}$	NORESP indicates that an RT has either not responded or has responded later than the BC No Response Timeout time. The Enhanced Mini-ACE's No Response Timeout Time is defined per MIL-STD-1553B as the time from the mid-bit crossing of the parity bit to the mid-sync crossing of the RT Status Word. The value of the No Response Timeout value is programmable from among the nominal values 18.5, 22.5, 50.5, and 130 µs (±1 µs) by means of bits 10 and 9 of Configuration Register #5.
0009	FMT ERR	$\overline{\text{FMT ERR}}$	FMT ERR indicates that the received portion of the most recent message contained one or more violations of the 1553 message validation criteria (sync, encoding, parity, bit count, word count, etc.), or the RT's status word received from a responding RT contained an incorrect RT address field.
000A	GD BLK XFER	$\overline{\text{GD BLK XFER}}$	For the most recent message, GD BLK XFER will be set to logic "1" following completion of a valid (error-free) RT-to-BC transfer, RT-to-RT transfer, or transmit mode code with data message. This bit is set to logic "0" following an invalid message. GOOD DATA BLOCK TRANSFER is **always** logic "0" following a BC-to-RT transfer, a mode code with data, or a mode code without data. The Loop Test has **no effect** on GOOD DATA BLOCK TRANSFER. GOOD DATA BLOCK TRANSFER may be used to determine if the transmitting portion of an RT-to-RT transfer was error free.
000B	MASKED STATUS BIT	$\overline{\text{MASKED STATUS BIT}}$	Indicates that one or both of the following conditions have occurred for the most recent message: (1) If one (or more) of the Status Mask bits (14 through 9) in the BC Control Word is logic "0" **and** the corresponding bit(s) is (are) set (logic "1") in the received RT Status Word. In the case of the RESERVED BITS MASK (bit 9) set to logic "0," any or all of the 3 Reserved Status bits being set will result in a MASKED STATUS SET condition; and/or (2) If BROADCAST MASK ENABLED/$\overline{\text{XOR}}$ (bit 11 of Configuration Register #4) is logic "1" **and** the MASK BROADCAST bit of the message's BC Control Word is logic "0" and the BROADCAST COMMAND RECEIVED bit in the received RT Status Word is logic "1."
000C	BAD MESSAGE	GOOD MESSAGE	Indicates either a format error, loop test failure, or no response error for the most recent message. Note that a "Status Set" condition has **no effect** on the "BAD MESSAGE/GOOD MESSAGE" condition code.
000D	RETRY0	$\overline{\text{RETRY0}}$	These two bits reflect the retry status of the most recent message. The number of times that the message was retried is delineated by these two bits as shown below:
000E	RETRY1	$\overline{\text{RETRY1}}$	
000F	ALWAYS	NEVER	The ALWAYS flag should be set (bit 4 = 0) to designate an instruction as unconditional. The NEVER bit (bit 4 = 1) can be used to implement an NOP or “skip” instruction.

RETRY COUNT 1 (BIT 14)	RETRY COUNT 0 (BIT 13)	NUMBER OF MESSAGE RETRIES
0	0	0
0	1	1
1	0	N/A
1	1	2

The Enhanced Mini-ACE core BC message sequence control capability enables a high degree of offloading the host processor. This includes using the various timing functions to enable autonomous structuring of major and minor frames. In addition, by implementing conditional jumps and subroutine calls, the message sequence control processor greatly simplifies the insertion of asynchronous, or "out-of-band" messages.

EXECUTE AND FLIP OPERATION

The BC's XQF, or "Execute and Flip" operation, provides some unique capabilities. Following execution of this unconditional instruction, if the condition code tests TRUE, the BC will modify the value of the current XQF instructions pointer parameter by toggling bit 4 in the pointer. That is, if the selected condition flag tests true, the value of the parameter will be updated to the value = old address XOR 0x0010. As a result, the next time that this line in the instruction list is executed, the Message Control/Status Block at the updated address will be processed. The operation of the XQF instruction is illustrated in FIGURE 4.

There are multiple ways of utilizing the "execute and flip" functionality. One is to facilitate the implementation of a double buffering data scheme for individual messages. This allows the message sequence control processor to "ping-pong" between a pair of data buffers for a particular message. By doing so, the host processor can access one of the two Data Word Blocks, while the BC reads or writes the alternate Data Word Block.

GENERAL PURPOSE QUEUE

The ACE-Core BC allows for the creation of a general purpose queue. This data structure provides mechanisms to push various items on this queue. These include the contents of Time Tag Register, the Block Status Word for the most recent message, an immediate data value, or the contents of a specified memory address.

FIGURE 5 illustrates the operation of the BC General Purpose Queue. Note that the BC General Purpose Queue Pointer Register will always point to the next address location (modulo 64); that is, the location following the last location written by the BC message sequence control engine.

If enabled, a BC GENERAL PURPOSE QUEUE ROLLOVER interrupt will be issued when the value of the queue pointer address rolls over at a 64-word boundary.

REMOTE TERMINAL ARCHITECTURE

The EBR-1553 supports multi-RT functionality through an embedded multiplexer function. The EBR-1553 RT protocol will respond to up to four RT addresses on the four interfaces. The multiplexer function limits the RT to only "listen" on one port at any one time.

The ACE-Core RT protocol design implements all of the EBR-1553/MIL-STD-1553 message formats. The Enhanced Mini-ACE core RT performs comprehensive error checking, word and format validation. One of the main features of the ACE-Core RT is its choice of memory management options. These include single buffering by subaddress, double buffering for individual receive subaddresses, circular buffering by individual subaddresses, and global circular buffering for multiple (or all) subaddresses.

Other features of the ACE-Core RT include a set of interrupt conditions, an interrupt status queue with filtering based on valid

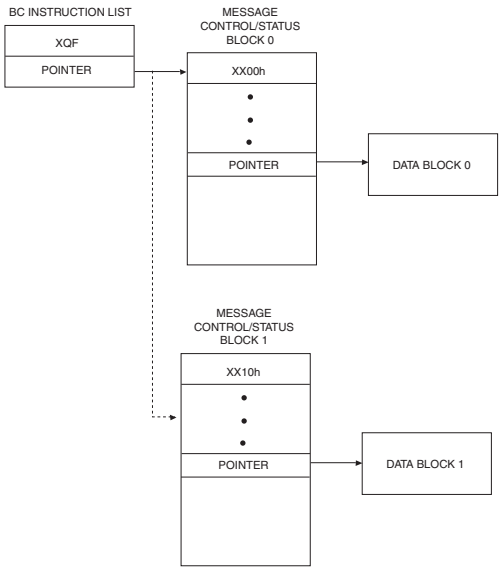


FIGURE 4. EXECUTE AND FLIP (XQF) OPERATION

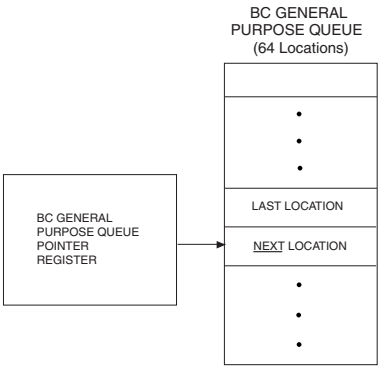


FIGURE 5. BC GENERAL PURPOSE QUEUE

and/or invalid messages, internal command illegalization, programmable busy by subaddress, multiple options on time tagging, and an “auto-boot” feature which allows the RT to initialize as an online RT with the busy bit set following power turn-on.

RT MEMORY MANAGEMENT

The ACE-Core provides a variety of RT memory management capabilities. The choice of memory management scheme is fully programmable on a transmit/receive/broadcast subaddress basis.

For each transmit or receive subaddress either a single-message data block, a double-buffered configuration (two alternating Data Word Blocks), or a variable sized (128 to 8192 words) subaddress circular buffer may be allocated for data storage. The memory management scheme for individual subaddresses is designated by means of the subaddress control word.

For received data, there is also a global circular buffer mode. In this configuration, the data words received from multiple (or all) subaddresses are stored in a common circular buffer structure. Like the subaddresses circular buffer the size of the global circular buffer is programmable with a range of 128 to 8192 data words.

The double buffering feature provides a means for the host processor to easily access the most recent, complete received block of valid Data Words for any given subaddress. In addition to helping ensure data sample consistency, the double buffer options provide a means of greatly reducing host processor overhead for multi-message bulk data transfer applications.

End-of-message interrupts may be enabled either globally (following all messages), following error messages, on a transmit or receive subaddress or mode code basis, or when a circular buffer reaches its midpoint (50% rollover boundary) or lower (100% boundary). A pair of interrupt status registers allow the host processor to determine the cause of all interrupts by means of a single read operation.

SINGLE-BUFFERED MODE

The operation of the single-buffered RT mode is illustrated in FIGURE 6. In the single-buffered mode the respective lookup table entry must be written by the host processor. Received data words are written to, or transmitted data words are read from the data word block with starting address referenced by the lookup table pointer. In the single-buffered mode, the current lookup table pointer is not updated by the ACE-Core memory management logic. Therefore, if a subsequent message is received for the same subaddress, the same data word block will be overwritten or over-read.

SUBADDRESS DOUBLE-BUFFERING MODE

The ACE-Core provides a double-buffering mechanism for received data that may be selected on an individual subaddress basis for any and all receive subaddresses. This is illustrated in FIGURE 7. It should be noted that the Subaddress Double Buffering mode is applicable for receive data only, not for transmit data. Double buffering of transmit messages may be easily implemented by software techniques.

The purpose of the subaddress double buffering-mode is to provide data sample consistency to the host processor. This is accomplished by allocating two 32-word data word blocks for each individual receive subaddress. At any given time, one of the blocks will be designated as the “active” 1553 block while the other will be considered as “inactive”. The data words for the next receive command to that subaddress will be stored in the active block. Following receipt of a valid message, the ACE-Core will automatically switch the active and inactive blocks for that subaddress. As a result, the latest, valid, complete data block is always accessible to the host processor.

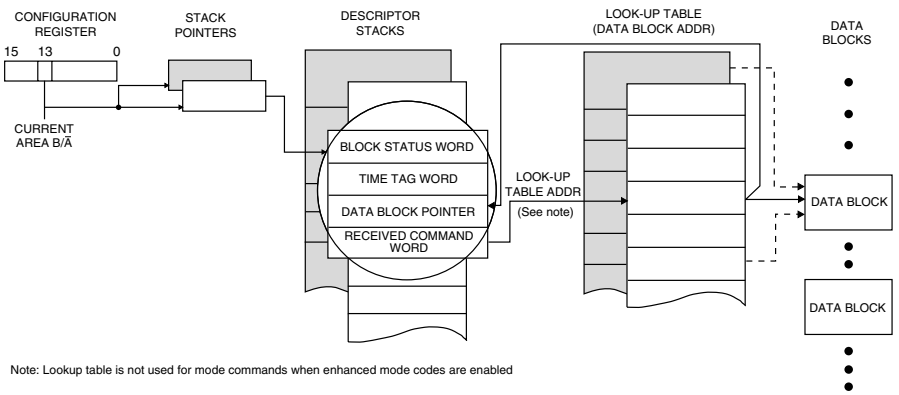


FIGURE 6. RT SINGLE BUFFERED MODE

CIRCULAR BUFFER MODE

The operation of the ACE-Core's circular buffer RT memory management mode is illustrated in FIGURE 8. As in the single buffered and double-buffered modes, the individual lookup table entries are initially loaded by the host processor. At the start of each message, the lookup table entry is stored in the third position of the respective message block descriptor in the descriptor stack area of RAM. Receive or transmit data words are transferred to (from) the circular buffer, starting at the location referenced by the lookup table pointer.

In general, the location after the last data word written or read (modulo the circular buffer size) during the message is written to the respective lookup table location during the end-of-message sequence. By so doing, data for the next message for the respective transmit receive subaddress will be accessed from the next lower contiguous block of locations in the circular buffer.

For the case of a receive (or broadcast receive) message with a data word error, there is an option such that the lookup table pointer will only be updated following receipt of a valid message. That is, the pointer will not be updated following receipt of a message with an error in a data word. This allows failed messages in a bulk data transfer to be retried without disrupting the circular buffer data structure, and without intervention by the RT's host processor.

GLOBAL CIRCULAR BUFFER

Beyond the programmable choice of single buffer mode, double buffer mode, or circular buffer mode, programmable on an individual subaddress basis, the ACE-Core RT architecture provides an additional option, a variable sized global circular buffer. The ACE-Core RT allows for a mix of single buffered, double buffered, and individually circular buffered subaddresses, along with the use of the global double buffer for any arbitrary group of receive subaddresses.

In the global circular buffer mode, the data for multiple receive subaddresses is stored in the same circular buffer data structure. The size of the global circular buffer may be programmed for 128, 256, 512, 1024, 2048, 4096, or 8192 words, by means of Configuration Register #6. Individual subaddresses may be mapped to the global circular buffer by means of their respective subaddress control words.

The pointer to the Global Circular Buffer is stored in location 0101 (for Area A), or location 0105 (for Area B).

The global circular buffer option provides a highly efficient method for storing received message data. It allows for frequently used subaddresses to be mapped to individual data blocks, while also providing a method for asynchronously received messages to infrequently used subaddresses to be logged to a com-

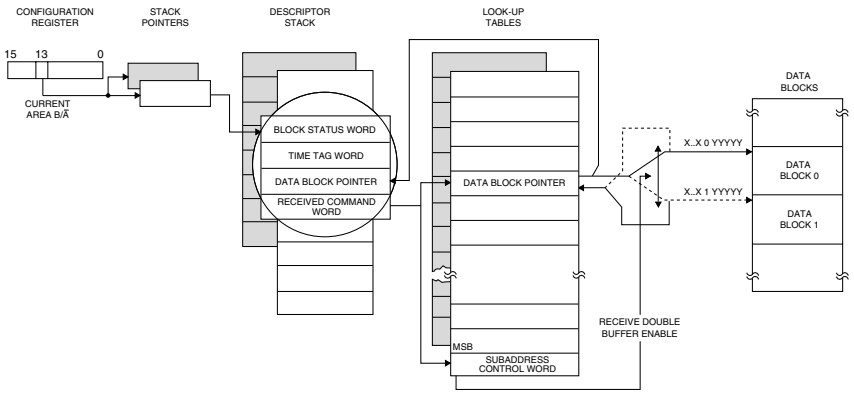


FIGURE 7. RT DOUBLE BUFFERED MODE

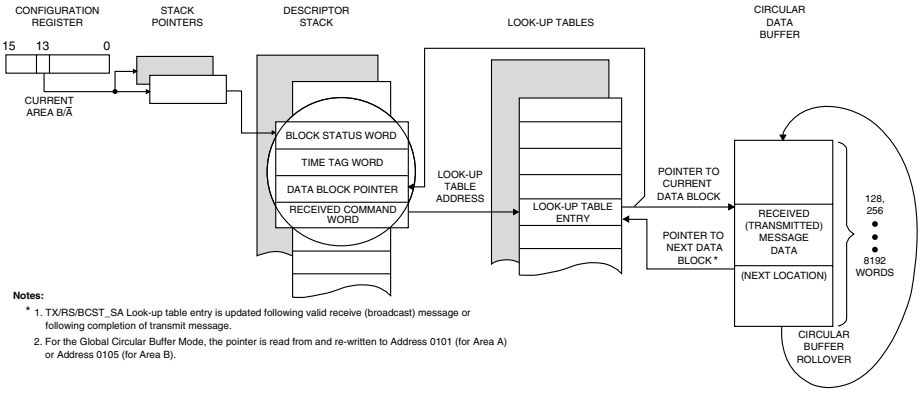


FIGURE 8. RT CIRCULAR BUFFERED MODE

mon area. Alternatively, the global circular buffer provides an efficient means for storing the received data words for all subaddresses. Under this method, all received data words are stored chronologically, regardless of subaddress.

RT DESCRIPTOR STACK

The descriptor stack provides a chronology of all messages processed by the ACE-Core RT. Reference FIGURE 6, FIGURE 7, and FIGURE 8. Similar to BC mode, there is a four-word block descriptor in the Stack for each message processed. The four entries to each block descriptor are the Block Status Word, Time Tag Word, the pointer to the start of the message's data block, and the 16-bit received Command Word.

The RT Block Status Word includes indications of whether a particular message is ongoing or has been completed, what bus channel it was received on, indications of illegal commands, and flags denoting various message error conditions. For the double buffering, subaddress circular buffering, and global circular buffering modes, the data block pointer may be used for locating the data blocks for specific messages. Note that for mode code commands, there is an option to store the transmitted or received data word as the third word of the descriptor, in place of the data block pointer.

The Time Tag Word provides a 16-bit indication of relative time for individual messages. The resolution of the ACE-Core's time tag is programmable from among 2, 4, 8, 16, 32, or 64 µs/LSB. There is also a provision for using an external clock input for the time tag (consult factory). If enabled, there is a time tag rollover interrupt, which is issued when the value of the time tag rolls over from 0xFFFF to 0. Other time tag options include the capabilities to clear the time tag register following receipt of a Synchronize (without data) mode command and/or to set the time tag following receipt of a Synchronize (with data) mode command. For the latter, there is an added option to filter the “set” capability based on the LSB of the received data word being equal to logic “0”.

RT INTERRUPTS

The ACE-Core offers a great deal of flexibility in terms of RT interrupt processing. By means of the ACE-Core's two Interrupt Mask Registers, the RT may be programmed to issue interrupt requests for the following events/conditions: End-of-(every) Message, Message Error, Selected (transmit or receive) Subaddress, 100% Circular Buffer Rollover, 50% Circular Buffer Rollover, 100% Descriptor Stack Rollover, 50% Descriptor Stack Rollover, Selected Mode Code, Transmitter Timeout, Illegal Command, and Interrupt Status Queue Rollover.

INTERRUPT FOR 50% ROLLOVERS OF STACKS, CIRCULAR BUFFERS

The ACE-Core RT and Monitor are capable of issuing host interrupts when a subaddress circular buffer pointer or stack pointer crosses its mid-point boundary. For RT circular buffers, this is applicable for both transmit and receive subaddresses. Reference FIGURE 9. There are four interrupt mask and interrupt status register bits associated with the 50% rollover function: (1) RT circular buffer; (2) RT command (descriptor) stack; (3) Monitor command (descriptor) stack; and (4) Monitor data stack.

The 50% rollover interrupt is beneficial for performing bulk data transfers. For example, when using circular buffering for a particular receive subaddress, the 50% rollover interrupt will inform the host processor when the circular buffer is half full. At that time, the host may proceed to read the received data words in the upper half of the buffer, while the ACE-Core RT writes received data words to the lower half of the circular buffer. Later, when the RT issues a 100% circular buffer rollover interrupt, the host can proceed to read the received data from the lower half of the

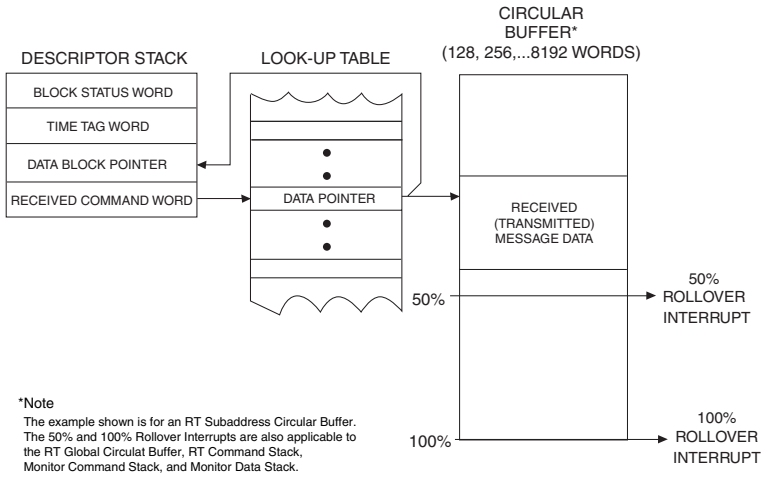


FIGURE 9. 50% AND 100% ROLLOVER INTERRUPTS

buffer, while the ACE-Core RT continues to write received data words to the upper half of the buffer.

INTERRUPT STATUS QUEUE

The ACE-Core RT, Monitor, and combined RT/Monitor modes include the capability for generating an interrupt status queue. As illustrated in FIGURE 10, this provides a chronological history of interrupt generating events and conditions. In addition to the Interrupt Mask Register, the Interrupt Status Queue provides additional filtering capability such that only valid messages and/or only invalid messages may result in the creation of an entry to the Interrupt Status Queue. The pointer to the Interrupt Status Queue is stored in the INTERRUPT VECTOR QUEUE POINTER REGISTER (register address 1F). This register must be initialized by the host, and is subsequently incremented by the RT message processor. The interrupt status queue is 64 words deep, providing the capability to store entries for up to 32 messages.

The queue rolls over at addresses of modulo 64. The events that result in queue entries include both message-related and non-message related events. Note that the Interrupt Vector Queue Pointer Register will always point to the next location (modulo 64) following the last vector/pointer pair written by the ACE-Core RT, Monitor, or RT/Monitor.

Each event that causes an interrupt results in a two-word entry to be written to the queue. The first word of the entry is the interrupt vector. The vector indicates which interrupt event(s)/condition(s) caused the interrupt.

The interrupt events are classified into two categories: message interrupt events and non-message interrupt events. Message-based interrupt events include End-of-Message, Selected mode code, Format error, Subaddress control word interrupt, RT Circular buffer Rollover, Handshake failure, RT Command stack rollover, transmitter timeout, and RT Circular buffer 50% rollover. Non-message interrupt events/conditions include Time tag rollover, RT address parity error, MT data stack rollover, MT command stack rollover, RAM parity error, RT Command stack 50% rollover, MT data stack 50% rollover, MT command stack 50% rollover, and BIT completed.

Bit 0 of the interrupt vector (interrupt status) word indicates whether the entry is for a message interrupt event (if bit 0 is logic "1") or a non-message interrupt event (if bit 0 is logic "0"). It is not possible for one entry on the queue to indicate both a message interrupt and a non-message interrupt.

As illustrated in FIGURE 10, for a message interrupt event, the parameter word is a pointer. The pointer will reference the first word of the RT or MT command stack descriptor (i.e., the Block Status Word).

For the RT address Parity Error and Protocol Self-test Complete non-message interrupts, the parameter is not used; it will have a value of 0000. For Time Tag Rollover non-message interrupts, the parameter will be a pointer to the descriptor stack.

If enabled, an INTERRUPT STATUS QUEUE ROLLOVER interrupt will be issued when the value of the queue pointer address rolls over at a 64-word address boundary.

RT COMMAND ILLEGALIZATION

The ACE-Core provides an internal mechanism for RT Command Word illegalizing. By means of a 256-word area in shared RAM, the host processor may designate that any message be illegalized, based on the command word T/R bit, subaddress, and word count/mode code fields. The ACE-Core illegalization scheme provides the maximum in flexibility, allowing any subset of the 4096 possible combinations of own address, T/R bit, subaddress, and word count/mode code to be illegalized.

BUSY BIT

The ACE-Core RT provides two different methods for setting the Busy status word bit: (1) globally, by means of Configuration Register #1; or (2) on a T/R-bit/subaddress basis, by means of a RAM lookup table. If the host CPU asserts the BUSY bit low in Configuration Register #1, the ACE-Core RT will respond to all commands with the Busy bit set in its RT Status Word.

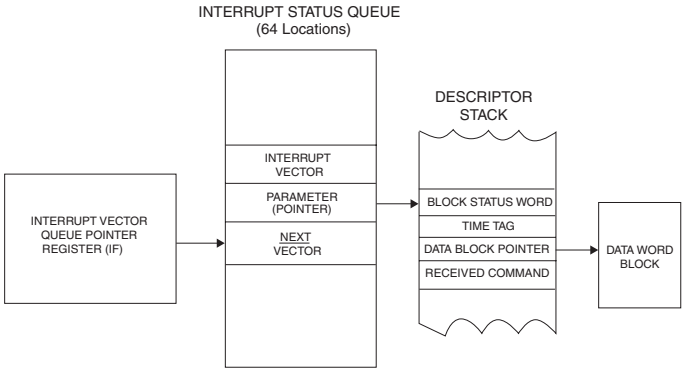


FIGURE 10. RT (AND MONITOR) INTERRUPT STATUS QUEUE (Shown for Message Interrupt Event)

TABLE 7. RT BIT WORD	
BIT	**DESCRIPTION**
15(MSB)	TRANSMITTER TIMEOUT
14	LOOP TEST FAILURE B
13	LOOP TEST FAILURE A
12	HANDSHAKE FAILURE
11	TRANSMITTER SHUTDOWN B
10	TRANSMITTER SHUTDOWN A
9	TERMINAL FLAG INHIBITED
8	BIT TEST FAILURE
7	HIGH WORD COUNT
6	LOW WORD COUNT
5	INCORRECT SYNC RECEIVED
4	PARITY / MANCHESTER ERROR RECEIVED
3	RT-to-RT GAP / SYNC / ADDRESS ERROR
2	RT-to-RT NO RESPONSE ERROR
1	COMMAND WORD CONTENTS ERROR
0(LSB)	DATA 0

Alternatively, there is a Busy lookup table in the ACE-Core shared RAM. By means of this table it is possible for the host processor to set the busy bit for any selectable subset of the 64 combinations of T/R bit and subaddress, and for transmit broadcast mode commands. If the busy bit is set for a transmit command, the ACE-Core RT will respond with the busy bit set in the status word, but will not transmit any data words. If the busy bit is set for a receive command, the RT will also respond with the busy status bit set. There are two programmable options regarding the reception of data words for a non-mode code receive command for which the RT is busy: (1) to transfer the received data words to shared RAM; or (2) to not transfer the data words to shared RAM.

RT ADDRESS

The design of the BU-6558XCX Series supports the Miniature Munitions / Store Interface specification for assigning the RT address via the CANBus serial interface.

RT BUILT-IN TEST (BIT) WORD

The bit map for the ACE-Core's internal RT Built-in-Test (BIT) Word is indicated in TABLE 7.

OTHER RT FEATURES

The ACE-Core includes options for the Terminal flag status word bit to be set either under software control and/or automatically following a failure of the loopback self-test. Other software programmable RT options include software programmable RT status and RT BIT words, automatic clearing of the Service Request bit following receipt of a Transmit vector word mode command, options regarding Data Word transfers for the Busy and Message error (illegal) Status word bits, and options for the handling of 1553A and reserved mode codes.

MONITOR ARCHITECTURE

The ACE-Core includes three monitor modes:

1) A Word Monitor mode.

2) A selective message monitor mode.

3) A combined RT/message monitor mode.

For new applications, it is recommended that the selective message monitor mode be used, rather than the word monitor mode. Besides providing monitor filtering based on RT address, T/R bit, and subaddress, the message monitor eliminates the need to determine the start and end of messages by software.

WORD MONITOR MODE

In the Word Monitor Terminal mode, the ACE-Core monitors both 1553 buses. After the software initialization and Monitor Start sequences, the ACE-Core stores all Command, Status, and Data Words received from both buses. For each word received from either bus, a pair of words is stored to the ACE-Core's shared RAM. The first word is the word received from the 1553 bus. The second word is the Monitor Identification (ID), or "Tag" word. The ID word contains information relating to bus channel, word validity, and inter-word time gaps. The data and ID words are stored in a circular buffer in the shared RAM address space.

SELECTIVE MESSAGE MONITOR MODE

The ACE-Core Selective Message Monitor provides monitoring of 1553 messages with filtering based on RT address, T/R bit, and subaddress with no host processor intervention. By autonomously distinguishing between 1553 command and status words, the Message Monitor determines when messages begin and end, and stores the messages into RAM, based on a programmable filter (RT address, T/R bit, and subaddress).

The selective monitor may be configured as just a monitor, or as a combined RT/Monitor. In the combined RT/Monitor mode, the ACE-Core functions as an RT for one RT address, and as a selective message monitor for the other 30 RT addresses. The ACE-Core Message Monitor contains two stacks, a command stack and a data stack, that are independent from the BC/RT command stack. The pointers for these stacks are located at fixed locations in the RAM.

MONITOR SELECTION FUNCTION

Following receipt of a valid command word in Selective Monitor mode, the ACE-Core will reference the selective monitor lookup table to determine if this particular command is enabled. The address for this location is determined by means of an offset based on the RT Address, T/R bit, and Subaddress bit 4 of the current command word, and concatenating it to the monitor

lookup table base address of 0500-0501 (hex). The bit location within this word is determined by subaddress bits 3-0 of the current command word.

If the specified bit in the lookup table is logic “0”, the command is not enabled, and the ACE-Core will ignore this command. If this bit is logic “1”, the command is enabled and the ACE-Core will create an entry in the monitor command descriptor stack (based on the monitor command stack pointer), and store the data and status words associated with the command into sequential locations in the monitor data stack. In addition, for an RT-to-RT transfer in which the receive command is selected; the second command word (the transmit command) is stored in the monitor data stack.

SELECTIVE MESSAGE MONITOR MEMORY ORGANIZATION

FIGURE 11 illustrates the Selective Message Monitor operation. Upon receipt of a valid Command Word, the ACE-Core will reference the Selective Monitor Lookup Table to determine if the current command is enabled. If the current command is disabled, the ACE-Core monitor will ignore (and not store) the current message. If the command is enabled, the monitor will create an entry in the Monitor Command Stack at the address location referenced by the Monitor Command Stack Pointer, and an entry in the monitor data stack starting at the location referenced by the monitor data stack pointer.

The format of the information in the data stack depends on the format of the message that was processed. For example, for a BC-to-RT transfer (receive command) the monitor will store the command word in the monitor command descriptor stack, with the data words and the receiving RT's status word stored in the monitor command stack.

The size of the monitor command stack is programmable, with choices of 256, 1K, 4K, or 16K words. The monitor data stack size is programmable with choices of 512, 1K, 2K, 4K, 8K, 16K, 32K or 64K words.

MONITOR INTERRUPTS

Selective monitor interrupts may be issued for End-of-message and for conditions relating to the monitor command stack pointer and monitor data stack pointer. The latter, which are shown in FIGURE 9, include Command Stack 50% Rollover, Command Stack 100% Rollover, Data Stack 50% Rollover, and Data Stack 100% Rollover.

The 50% rollover interrupts may be used to inform the host processor when the command stack or data stack is half full. At that time, the host may proceed to read the received messages in the upper half of the respective stack, while the ACE-Core monitor writes messages to the lower half of the stack. Later, when the monitor issues a 100% stack rollover interrupt, the host can proceed to read the received data from the lower half of the stack, while the ACE-Core monitor continues to write received data words to the upper half of the stack.

INTERRUPT STATUS QUEUE

Like the ACE-Core RT, the Selective Monitor mode includes the capability for generating an interrupt status queue. As illustrated in FIGURE 10, this provides a chronological history of interrupt generating events. Besides the two Interrupt Mask Registers, the Interrupt Status Queue provides additional filtering capability, such that only valid messages and/or only invalid messages may result in entries to the Interrupt Status Queue. The interrupt status queue is 64 words deep, providing the capability to store entries for up to 32 monitored messages.

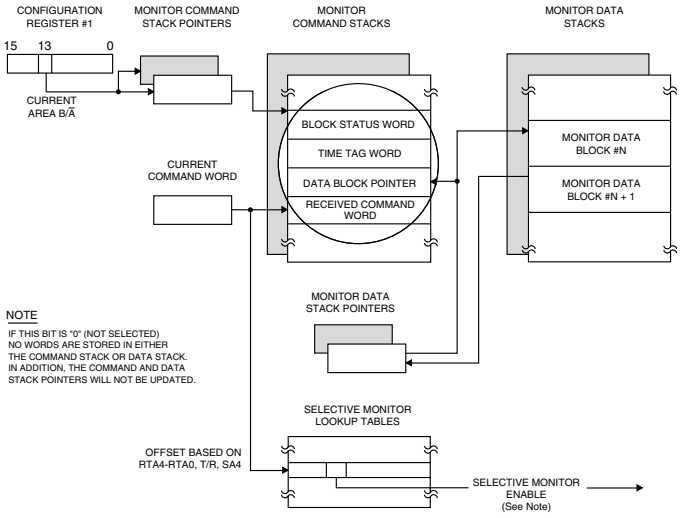


FIGURE 11. SELECTIVE MESSAGE MONITOR MEMORY MANAGEMENT

TIME TAG

The ACE-Core includes an internal read/writable Time Tag Register. This register is a CPU read/writable 16-bit counter with a programmable resolution of either 2, 4, 8, 16, 32, or 64 µs per LSB. Another option allows software-controlled incrementing of the Time Tag Register. This supports self-test for the Time Tag Register. For each message processed, the value of the Time Tag Register is loaded into the second location of the respective descriptor stack entry (“TIME TAG WORD”) for both the BC and RT mode.

The functionality involving the Time Tag Register that's compatible with ACE/Mini-ACE (Plus) includes: the capability to issue an interrupt request and set a bit in the Interrupt Status Register when the Time Tag Register rolls over FFFF to 0000; for RT mode, the capability to automatically clear the Time Tag Register following reception of a Synchronize (without data) mode command, or to load the Time Tag Register following a Synchronize (with data) mode command.

Additional time tag features supported by the ACE-Core include the capability for the BC to transmit the contents of the Time Tag Register as the data word for a Synchronize (with data) mode command; the capability for the RT to “filter” the data word for the Synchronize with data mode command, by only loading the Time Tag Register if the LSB of the received data word is “0”; an instruction enabling the BC Message Sequence Control engine to autonomously load the Time Tag Register; and an instruction enabling the BC Message Sequence Control engine to write the value of the Time Tag Register to the General Purpose Queue.

INTERRUPTS

The ACE-Core series components provide many programmable options for interrupt generation and handling. Individual interrupts are enabled by the two Interrupt Mask Registers (#1 or #2). The host processor may easily determine the cause of the interrupt by using the Interrupt Status Register (#1 or #2). The two Interrupt Status Registers (#1 and #2) provide the current state of the interrupt conditions. The Interrupt Status Registers may be updated in two ways. In the one interrupt handling mode, a particular bit in the Interrupt Status Register (#1 or #2) will be updated only if the event occurs and the corresponding bit in the Interrupt Mask Register (#1 or #2) is enabled. In the Enhanced interrupt handling mode, a particular bit in the Interrupt Status Register (#1 or #2) will be updated if the condition exists regardless of the contents of the corresponding Interrupt Mask Register bit. In any case, the respective Interrupt Mask Register (#1 or #2) bit enables an interrupt for a particular condition.

The ACE-Core supports all the interrupt events from ACE/Mini-ACE (Plus), including Transmitter Timeout, BC/RT Command Stack Rollover, MT Command Stack and Data Stack Rollover, Handshake Error, BC Retry, RT Address Parity Error, Time Tag Rollover, RT Circular Buffer Rollover, BC Message, RT Subaddress, BC End-of-Frame, Format Error, BC Status Set, RT Mode Code, MT Trigger, and End-of-Message.

In the ACE-Core's Enhanced BC mode, there are four user-defined interrupt bits. The BC Message Sequence Control Engine includes an instruction enabling it to issue these interrupts at any time.

For RT and Monitor modes, the ACE-Core architecture includes an Interrupt Status Queue. This provides a mechanism for logging messages that result in interrupt requests. Entries to the Interrupt Status Queue may be filtered such that only valid and/or invalid messages result in entries on the queue.

The ACE-Core incorporates additional interrupt conditions beyond ACE/Mini-ACE (Plus), based on the addition of Interrupt Mask Register #2 and Interrupt Status Register #2. This is accomplished by chaining of the two Interrupt Status Registers (#1 and #2) using one of the bits in Interrupt Status Register #2 to indicate an interrupt has occurred in Interrupt Status Register #1. Additional interrupts include “Self Test Completed”, masking bits for the Advanced BC Control Interrupts, 50% Rollover interrupts for RT Command Stack, RT Circular Buffers, MT Command Stack, and MT Data Stack; BC Op Code Parity Error, (RT) Illegal Command, (BC) General Purpose Queue or (RT/MT) Interrupt Status Queue Rollover, Call Stack Pointer Register Error, BC Trap Op Code, and four User-Defined interrupts for the Enhanced BC mode.

TABLE 8. MEMORY MAPPED REGISTERS - 1K BYTE (512 WORD)	
BYTE ADDRESS (HEX)	**REGISTER(S)**
(0000 — 007F)	EBR-1553 BC/RT Registers
(0080 — 00FF)	MIL-STD-1553 BC/MT/RT Registers
(0100 — 011F)	RESERVED
(0120 — 013F)	
(0140 — 015F)	
(0160 — 017F)	
-180	MSCI Serial Address Register 0
-182	MSCI Serial Address Register 1
-184	Store RT Address Register 0
-186	Store RT Address Register 1
(0188 — 019F)	RESERVED
(01A0)	Card ID Register
(01A2)	Channel/Segment Select Register
(01A4)	Interrupt Status Register
(01A6)	Discrete I/O Register 0
(01A8)	Discrete I/O Register 1
(01AA — 01AF)	EBR to CANBus Loop Back Test Registers
(01B0 — 03FF)	Spare (592 bytes (128h words))

TABLE 9. CONNECTOR P1 PINOUTS	
PIN	**FUNCTION**
1	CANH
2	CANL
3	
4	
5	CHASSIS GROUND
6	
7	
8	
9	
10	GROUND

TABLE 10. TRIAX CONNECTOR J2 PINOUTS	
PIN	**FUNCTION**
Center	Signal
Outer	Signal_
Shield	Chassis Ground

TABLE 11. I/O CONNECTOR P5 PINOUTS	
PIN	**FUNCTION**
1	GROUND
2	DIO(1)
3	DIO(2)
4	DIO(3)
5	DIO(4)
6	DIO(5)
7	DIO(6)
8	DIO(7)
9	DIO(8)
10	DIO(9)
11	DIO(10)
12	DIO(11)
13	DIO(12)
14	GROUND

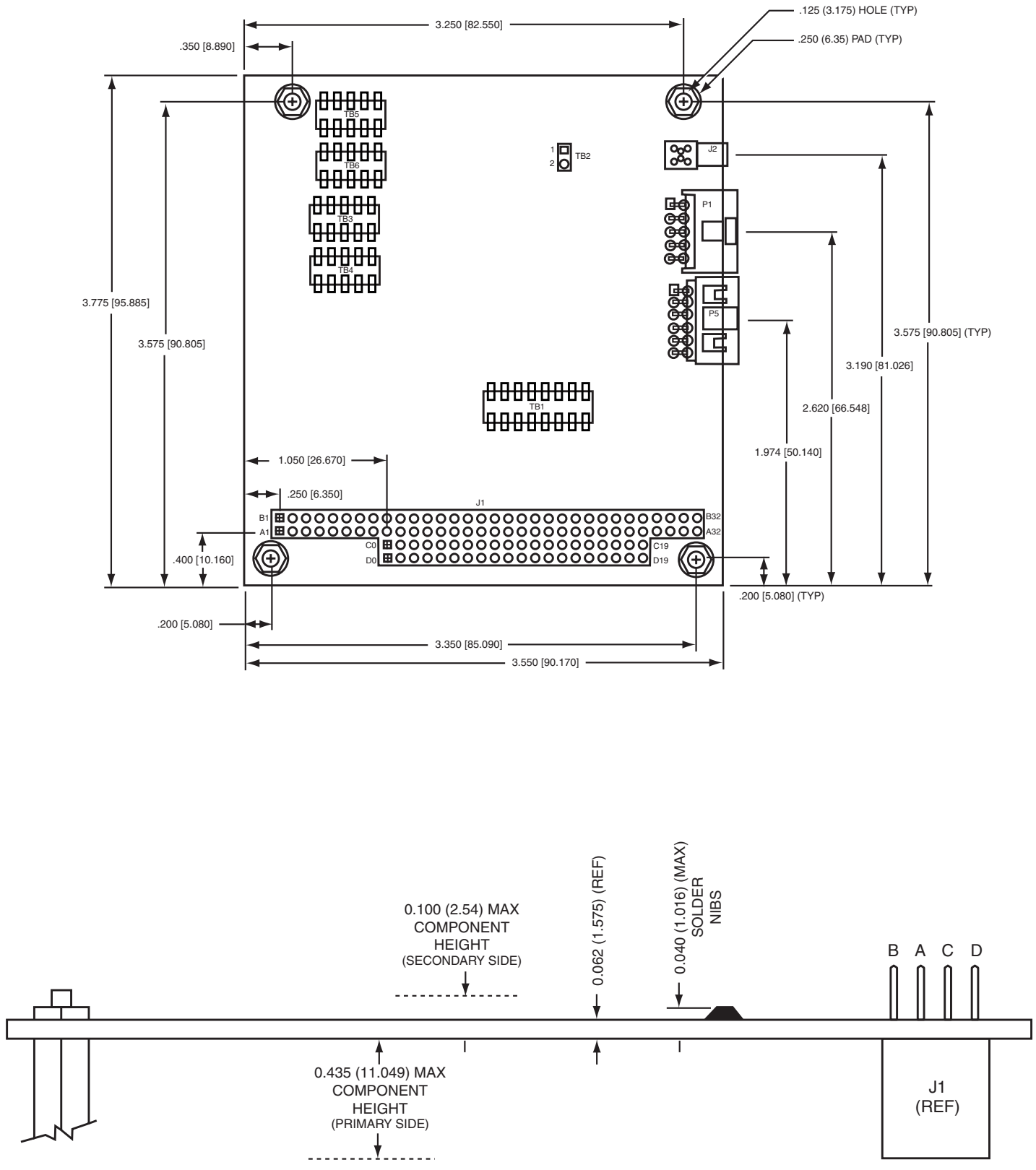


FIGURE 12. BU-6558XCX MECHANICAL OUTLINE

ORDERING INFORMATION

BU-6558XCX-X00N

Supplemental Process Requirements:
N = Conformal Coat
Blank = None

Temperature Range:
2 = -40°C to +71°C
3 = 0°C to +55°C

Number of EBR-1553 Ports:
1 = One
4 = Four

Card Type:
C = PC/104

Optional MIL-STD-1553:
0 = None
1 = MIL-STD-1553 Dual Redundant Channel Included

Model Number:
6558 = EBR-1553 Card

Note :
1. The above product contains tin-lead solder.

INCLUDED SOFTWARE:

BUS-6908XS0

ACE Runtime Library, Drivers / ACE Menu:
0 = DOS
3 = Windows NT/2000/XP
5 = Windows NT/2000/XP ACE Menu GUI

BU-69090SX

Enhanced Mini-ACE Runtime Library:
2 = VxWorks

DISCRETE MODULES / PC BOARD ASSEMBLIES PROCESSING TABLE

STANDARD DDC PROCESSING FOR DISCRETE MODULES/PC BOARD ASSEMBLIES		
TEST	**METHOD(S)**	**CONDITION(S)**
INSPECTION / WORKMANSHIP	IPC-A-610	Class 3
ELECTRICAL TEST	DDC ATP	—

NOTES:

Please visit our web site at **www.ddc-web.com** for the latest information.

©Data Device Corporation
105 Wilbur Place, Bohemia, New York, U.S.A. 11716-2482

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
Ireland - Tel: +353-21-341065, Fax: +353-21-341568
France - Tel: +33-(0)1-41-16-3424, Fax: +33-(0)1-41-16-3425
Germany - Tel: +49-(0)89-150012-11, Fax: +49-(0))89-150012-22
Japan - Tel: +81-(0)3-3814-7688, Fax: +81-(0)3-3814-7689
World Wide Web - http://www.ddc-web.com

DATA DEVICE CORPORATION
REGISTERED TO ISO 9001:2000
FILE NO. A5976