Model 2915

PCI Interface to 3922 Controller

INSTRUCTION MANUAL

February 5, 1999

(C) 1995, 1996, 1997, 1998, 1999
Copyright by
KineticSystems Company, LLC
Lockport, Illinois
All rights reserved

NOTICE

The 2x15 device may have trouble loading in some Pentium based computers.

If you experience trouble loading the KSC device driver:

1. Reboot
2. Enter the CMOS setup program
3. Enable the option asking if you are using a P&P operating system
4. Reboot the machine
5. Load the driver

This should take care of any loading problems.

KSC Support Staff

*****Special Option*****

Model 2915-S001

PCI Interface to 3922 Controller

September, 1996

© 1996
Copyright by
KineticSystems Corporation
Lockport, Illinois
All rights reserved

*****Special Option*****

Model 2915-S001

The Model 2915-S001 is the same as the Model 2915-Z1A except that it has been modified to stop scanning in Q-SCAN mode if scan reaches an open slot ($\overline{Q}$, $\overline{X}$ condition).

CONTENTS

FIGURES

PCI Interface for the 3922 Controller

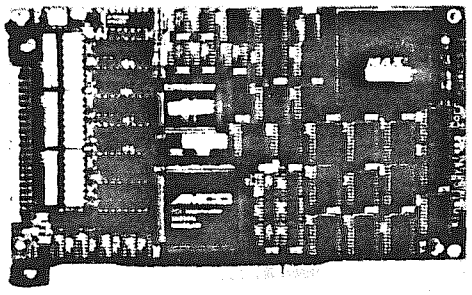
Allows a computer PCI bus to host up to eight 3922s **2915**

Features

- **Provides dedicated PCI interface**
- **Used with 3922 parallel-bus crate controllers**
- **Supports DMA transfers**
- **Controls up to eight CAMAC crates**
- **Bus lengths up to 90 meters (300 feet)**
- **Mounts in any PCI-compatible expansion slot**
- **Interrupts for CAMAC Done and LAMs**
- **RS-485 balanced-line signaling between the 2915 and 3922s for high noise immunity**
- **PCI configuration registers for programmable address selection**
- **Throughput up to 1 megabyte per second**

Typical Applications

- **Interfacing CAMAC to PCI computers**
- **General purpose data acquisition and control**
- **Laboratory automation**
- **Industrial process control**



General Description *(Product specifications and descriptions subject to change without notice.)*

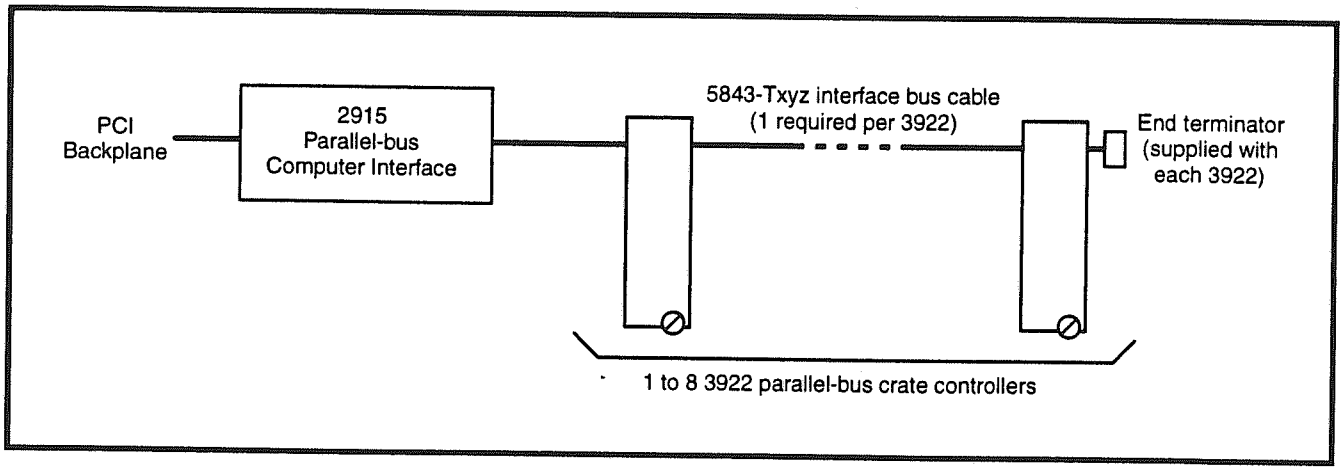
The 2915 is a computer-bus adapter for use with computers incorporating the Peripheral Component Interconnect (PCI) local bus. It mounts in any available PCI slot supporting 32-bit data transfers and communicates with the host using the 32-bit data path.

The 2915 supports up to eight 3922 Crate Controllers. Connection is made via a parallel bus that is a 40-conductor twisted-pair ribbon cable. Bus lengths must be ordered separately. Signaling on the parallel bus is accomplished using RS-485 balanced line drivers and receivers, which gives high noise immunity and allows an overall cable distance between the 2915 and the last 3922 of up to 90 m (300 feet). The last 3922 on the parallel bus is terminated with a termination card provided with the 3922.

Program transfers as well as Direct Memory Access (DMA) are supported by the 2915. The DMA transfers provide a speed-efficient mechanism to move large blocks of data to and from a 3922. DMA transfer modes include Q-Stop, Q-Ignore, Q-Repeat, and Q-Scan. Interrupt capability is also provided. An interrupt can be generated by "CAMAC Done" or by a pending Look-At-Me (LAM). In response to a LAM interrupt, the host performs a parallel poll operation to the 3922 via the 2915. During the parallel poll, each 3922 requesting service asserts one of the eight data lines (i.e., Crate 5 asserts Data line 5). This method allows efficient servicing of interrupts in multicrate systems.

The 2915 contains PCI configuration registers mandated by the PCI specification. These configuration registers permit the 2915 to be installed in a system without preselecting any address switches or straps.

Typical Application with the 3922 Crate Controller



Ordering Information

Model 2915-Z1A PCI interface to the model 3922 crate controller, with DMA

Associated Products

Model 3922-Z1B	Parallel Bus Crate Controller
Model AD3Z-NPA1	Software, Device Driver, Windows NT
Model AD3Z-VAA1	Software, Device Driver, Open VMS/Alpha AXP
Model 5843-Txyz	Bus Interface Cable (one required for each 3922)

INSTALLATION

The Model 2915 is designed to fit into any PCI slot that can accommodate a half-size card. To install the 2915, remove the cover of the computer that allows access to the PCI expansion slots. Locate an empty PCI expansion slot and remove the blank panel from the mounting rail. Insert the 2915 into the slot and secure it with the screw that was removed from the blank panel. Replace the cover of the computer.

Insert one end of the 40-conductor twisted pair ribbon cable into the 40-position header located on the rear of the 2915. Connect the other end of the cable to the first 3922. Refer to the 3922 instruction manual for additional connections made to the 3922.

CONFIGURATION SPACE

The PCI Specification mandates a 64-byte Configuration Header that describes the requirements of add-in cards. The data contained in this region uniquely identifies the device and allows for generic control of the device. The configuration data indicates the memory requirements of the device along with other device specific information.

This section describes the 64 bytes of configuration space implemented by the 2915. The following diagram is a composite chart showing the configuration header.

31 16	15 00	
Device Identification	Vendor Identification	00
Status	Command	04
Class Code	Revision	08
BIST / Header Type	Latency / Cache Size	0C
Base Address Register #1		10
Base Address Register #2		14
Base Address Register #3		18
Base Address Register #4		1C
Base Address Register #5		20
Base Address Register #6		24
Reserved		28
Reserved		2C
Expansion ROM Base Address		30
Reserved		34
Reserved		38
Maximum Latency / Minimum Latency	Interrupt Pin / Interrupt Line	3C

VENDOR IDENTIFICATION Field

The VENDOR IDENTIFICATION field contains read-only bits which identify the manufacturer of the device. The ID assigned to KineticSystems is 11F4 Hex.

DEVICE IDENTIFICATION Field

The DEVICE IDENTIFICATION field contains read-only bits which identify a particular device. The DEVICE ID field for this unit is 2915 Hex.

PCI COMMAND Register

The COMMAND field contains write/read bits used to configure basic PCI functions. The following diagram shows the COMMAND field as implemented by the 2915.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0	0	0	0	0	0	FST BTB	SER ENA	0	PER ENA	0	0	0	MAS ENA	MEM ENA	I/O ENA

<15:10> Not Used. These bits are not used and read as zeros.

<9> FAST BACK-TO-BACK is a write/read bit used to enable or disable the 2915 from executing "fast" back-to-back bus master cycles after completing a write cycle. Setting this bit to a one enables fast transfers and a zero disables the mode.

<8> SYSTEM ERROR ENABLE is a write/read bit used to enable and disable the 2915 from driving the PCI SYSTEM ERROR (SERR) signal. This signal is used by the 2915 to inform the host CPU of a parity error during an address or control portion of a bus operation. Setting this bit to a one enables the 2915 to assert SERR and disables with a zero.

<7> Not Used. These bits are not used and read as zeros.

<6> PARITY ERROR ENABLE is a write/read bit used to enable and disable the 2915 from driving the PCI PARITY ERROR (PERR) signal. This signal is asserted by the 2915 when a parity error is detected during a data transfer to/from the 2915. The PERR function is enabled by setting this bit to a one and disabled with a zero.

<5:3> Not Used. These bits are not used and read as zeros.

<2> BUS MASTER ENABLE is a write/read bit which enables and disables the 2915 from executing bus master operations. Setting this bit to a one enables the 2915 to function as a bus master and a zero disables the master operation.

<1> MEMORY SPACE ENABLE is a write/read bit that allows the 2915 to function in memory regions that may be defined in one of the base address registers. Since the 2915 is initially configured as an I/O device, this bit should be set to zero.

<0> I/O SPACE ENABLE is a write/read bit that allows the 2915 to function in I/O regions as defined in one of the base address registers. This bit is set to a one which allows the 2915 to function in I/O regions.

PCI STATUS Register

The PCI STATUS Register is used to record status information regarding PCI bus transfers. This register contains read-only bits and write/read bits. The following diagram shows the Status Register bits implemented by the 2915.

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
PAR DET	SIG SER	RCV MAB	RCV TAB	SIG TAB	DEV TM1	DEV TM0	DP RPT	1	0	0	0	0	0	0	0

<15> DETECTED PARITY ERROR is a write/read bit which is set whenever the 2915 detects a PCI parity error. This bit does not depend on the state of the PAR ENA bit in the PCI COMMAND Register. Once an error has been detected ,this may be cleared by writing a one to bit position 15.

<14> SIGNALED SYSTEM ERROR is a write/read bit which is set whenever the 2915 asserts the PCI SYSTEM ERROR (SERR) signal. Once this bit is set, it may be cleared by writing a one to bit position 14.

<13> RECEIVED MASTER ABORT is a write/read bit which is set when the 2915 is accessed as a target and the master aborts the transaction. This bit can be reset by writing a one to this bit position.

<12> RECEIVED TARGET ABORT is a write/read bit which is set when the 2915, acting as a bus master, has initiated a transfer and the addressed target aborts the transfer. This bit can be reset by writing a one to this bit position.

<11> SIGNALED TARGET ABORT is a write/read bit that is set when a bus master accesses the 2915 as a target and the 2915 aborts the cycle. This bit is reset when a one is written to this bit location.

<10:9> DEVICE SELECT TIMING 1 and 0 encode the timing of the PCI DEVSEL (Device Select) signal. This time reflects the slowest time that a device asserts DEVSEL for any bus command except Configuration Read and Configuration Write. Since the 2915 may be accesses in the fast mode, these bits are set to zero.

<8> DATA PARITY REPORTED is a write/read bit which is set when the 2915 detects a parity error when the 2915 is a bus master. This bit can be reset by writing a one to this bit position.

REVISION Field

The REVISION field contains read-only bits which reflect the current revision level of the 2915. The 2915 starts at revision one and subsequent revisions increment the number.

CLASS CODE Field

The CLASS CODE field actually contains three subfields that represent device characteristics. These three subfields are the BASE CLASS, the SUB-CLASS and the PROG I/F fields. These subfields define such parameters as network controllers, display controllers, video device, etc.

The 2915 does not fit into any of the defined class codes. Therefore, the class code that the 2915 uses is FF0000 Hex which indicates that the 2915 class code is undefined by the PCI specification.

CACHE LINE SIZE Field

The CACHE LINE SIZE field is used by the system to define the cache line size. The 2915 does not use Memory Write and Invalidates PCI bus cycles when operating as a bus master and therefore sets this field to zero.

LATENCY TIMER Register

The LATENCY TIMER REGISTER is only used when the 2915 is operating as a bus master. The value loaded in this register is the minimum number of PCI bus clocks that the 2915 can be guaranteed as a master. After the 2915 becomes bus master and asserts the PCI FRAME signal, the Latency Timer is decremented for each PCI bus clock. Subsequent to the timer decrementing to zero, the 2915 ignores the PCI bus grant signal and continues to transfer data until the timer expires. The value loaded into this register is in multiples of eight clock cycles since the low 3 bits of this field are hardwired to zero. This register is loaded with a value of F8 hex at power-up. If this register is written to via application software, the value of F8 must be maintained.

HEADER Field

The HEADER field establishes whether a PCI device contains a single function or multi-function PCI bus agent. Since the 2915 contains only a single function, this field is set to zero.

BUILD IN SELF TEST Field

This field is used to present Built In Self Test diagnostic results to a bus master. The 2915 does not implement BIST and returns a zero for this field.

BASE ADDRESS Registers

The BASE ADDRESS REGISTERS are used to specify the memory or I/O requirements of add-in devices and also to configure the base addresses of these devices.

After power-up, system software can determine how much address space a particular device requires by writing all ones to a base address register and then reading that value back. The device returns zeros in all address bit locations that do not define the base address.

The least significant bit in each of the base address registers is used for specifying the region of address space for which the device is to reside. A value of zero specifies a memory region and a value of one specifies an I/O region. The 2915 is configured to operate in the I/O region.

The 2915 implements two of the Base Address registers. The first Base Address register is used to communicate with the PCI Interface Operational Registers (IOR) and the second is used to communicate with the Parallel Bus Operational Registers (PBOR). The Interface Operational Registers require 16 longwords (64 bytes) of address space. A read of the first base address register after a write of all ones returns the value FFFFFFC1 Hex, indicating a request for 16 longwords of I/O space.

The Parallel Bus Operational Registers require 4 longwords (16 bytes) of address space. A read of the second base address register after a write of all ones returns the value FFFFFFF1 Hex, indicating a request for 4 longwords in I/O space.

After the system software has determined the total address space requirements of the system, it assigns the base addresses to memory and I/O devices by writing their Base Address Registers.

EXPANSION ROM BASE ADDRESS Register

This field is used to assign a physical memory address to expansion ROM in a system. The 2915 does not contain an expansion ROM and therefore does not require use of this field.

INTERRUPT LINE Field

This write/read field is used for communicating interrupt routing information and is configured by the PCI BIOS after power-up. The value in this field informs the system interrupt controller which pin of the controller the interrupt is connected to.

INTERRUPT PIN Field

The INTERRUPT PIN field is read-only and specifies which PCI interrupt pin that the 2915 is connected to. The 2915 returns a value of one in this field indicating that it uses the INTA interrupt.

MINIMUM GRANT Register

This write/read register is used by bus masters to specify the minimum amount of time the device needs for a period of burst transfers. Since the 2915 does not have this requirement, a value of zero must be used.

PCI INTERFACE OPERATIONAL Registers

The PCI Interface Operational Registers are contained in the PCI interface chip used on the 2915. This chip is the S5933 and is manufactured by Applied Micro Circuits Corporation. The base address of these registers is loaded by power-on BIOS routines and is contained in the Base Address Register #1 location of the PCI Configuration Registers.

Note: All references to Incoming and Outgoing are referred to the host. An Outgoing operation is a write operation from the host and an Incoming operation is a read operation from the 2915.

The following chart shows the various PCI Interface Registers along with their offsets from the base address.

Offset	Register	Access	
0	Outgoing Mailbox #1	W/R	*
4	Outgoing Mailbox #2	W/R	*
8	Outgoing Mailbox #3	W/R	*
C	Outgoing Mailbox #4	W/R	*
10	Incoming Mailbox #1	R	*
14	Incoming Mailbox #2	R	*
18	Incoming Mailbox #3	R	*
1C	Incoming Mailbox #4	R	*
20	Data FIFO	W/R	
24	Master Write Address	R	
28	Master Write Transfer Count	R	
2C	Master Read Address	R	
30	Master Read Transfer Count	R	
34	Mailbox Empty/Full Status	W/R	*
38	Interrupt Control/Status	W/R	
3C	Bus Master Control/Status	W/R	

* These registers are found in the PCI Interface Controller but not used by the 2915.

OUTGOING/INCOMING MAILBOXES

The Incoming and Outgoing Mailboxes are not used by the 2915.

DATA FIFO REGISTER

The Data FIFO Register is a write/read register located at an offset of 20 hex from the selected base address register #1 and is composed of two 8 x 32- bit FIFOs located in the PCI interface chip. When executing Parallel Bus operations, all write and read data passes through these FIFOs. Data transfers to or from these FIFOs can be done by either programmed transfers executed by the host or by allowing the 2915 to become a bus master and transfer the data.

Several status indicators are provided which indicate the amount of data contained in the PCI interface chip FIFOs. The Bus Master Control/Status register contains 6 status bits that correspond to the PCI Interface chip internal FIFOs.

When executing CAMAC operations, write and read data transferred during the operation pass through this FIFO register. CAMAC data words may be either 16 or 24-bits wide. 24-bit data words occupy one complete 32-bit FIFO word. When the selected CAMAC data word size is 16-bits, two data words are contained in each FIFO word. The following diagram shows how the CAMAC data words are packed in the FIFO register for 16-bit operations.

Offset 0 hex	CAMAC 16-Bit Data Word #2	CAMAC 16-Bit Data Word #1
Offset 4 hex	CAMAC 16-Bit Data Word #4	CAMAC 16-Bit Data Word #3
Offset 8 hex	CAMAC 16-Bit Data Word #6	CAMAC 16-Bit Data Word #5
Offset C hex	CAMAC 16-Bit Data Word #8	CAMAC 16-Bit Data Word #7

If, during 16-bit CAMAC read operations, an odd number of data words are requested to be transferred, the high order 16-bits of the last 32-bit FIFO data word are set to zero. During CAMAC write operations, the high order 8-bits are not used in the FIFO data word.

MASTER WRITE ADDRESS REGISTER

The Master Write Address Register is a write/read register located at an offset of 24 hex from the selected base address #1 and is used to specify the initial address accessed by the 2915 during DMA transfers from the 2915 to the PCI memory. A read of this register returns the last address that was accessed during a bus master write operation executed by the 2915. This address is actually pointing to one longword beyond the last address accessed since the memory address is incremented after each execution of a data transfer. All DMA transfers must begin on a longword boundary. The following diagram shows the bit pattern for the Master Write Address Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MWAR 31	MWAR 30	MWAR 29	MWAR 28	MWAR 27	MWAR 26	MWAR 25	MWAR 24	MWAR 23	MWAR 22	MWAR 21	MWAR 20	MWAR 19	MWAR 18	MWAR 17	MWAR 16
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MWAR 15	MWAR 14	MWAR 13	MWAR 12	MWAR 11	MWAR 10	MWAR 09	MWAR 08	MWAR 07	MWAR 06	MWAR 05	MWAR 04	MWAR 03	MWAR 02	0	0

<31:2> MASTER WRITE ADDRESS 31 through 2 are write/read bits used to specify the initial DMA address during block transfer operations. Reading this register returns the last address accessed during a DMA operation.

<1:0> These bits are not used since the DMA operations execute longword (32-bit) transfers.

MASTER WRITE TRANSFER COUNT REGISTER

The Master Write Transfer Count Register is a write/read register located at an offset of 28 hex from the selected base address #1 and is loaded with the number of bytes to transfer from the 2915 to PCI memory during a DMA operation. This counter is decremented by four for every PCI bus cycle executed, since the 2915 transfers only 32-bit data words during DMA. When this counter is decremented to zero, the DMA operation terminates and an interrupt may be optionally generated to the PCI bus. If a CAMAC block transfer operation is terminated due to an error, this register may be read to determine the number of data bytes not stored in PCI memory for the requested operation. The following diagram shows the bit pattern for the Master Write Transfer Count Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	MWTC 25	MWTC 24	MWTC 23	MWTC 22	MWTC 21	MWTC 20	MWTC 19	MWTC 18	MWTC 17	MWTC 16
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MWTC 15	MWTC 14	MWTC 13	MWTC 12	MWTC 11	MWTC 10	MWTC 09	MWTC 08	MWTC 07	MWTC 06	MWTC 05	MWTC 04	MWTC 03	MWTC 02	MWTC 01	MWTC 00

<31:26> These bits are not used and read as zeros.

<25:00> MASTER WRITE TRANSFER COUNT 25 through 0 are write/read bits used to specify the maximum number of bytes to transfer from the 2915 to PCI memory during DMA operations.

MASTER READ ADDRESS REGISTER

The Master Read Address Register is a write/read register located at an offset of 2C hex from the selected base address #1 and is used to specify the initial address accessed by the 2915 during DMA transfers from the PCI memory to the 2915. A read of this register returns the last address that was accessed during a bus master read operation executed by the 2915. This address is actually pointing to one longword beyond the last address accessed since the memory address is incremented after each execution of a data transfer. All DMA transfers must begin on a longword boundary. The following diagram shows the bit pattern for the Master Read Address Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MRAR 31	MRAR 30	MRAR 29	MRAR 28	MRAR 27	MRAR 26	MRAR 25	MRAR 24	MRAR 23	MRAR 22	MRAR 21	MRAR 20	MRAR 19	MRAR 18	MRAR 17	MRAR 16
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MRAR 15	MRAR 14	MRAR 13	MRAR 12	MRAR 11	MRAR 10	MRAR 09	MRAR 08	MRAR 07	MRAR 06	MRAR 05	MRAR 04	MRAR 03	MRAR 02	0	0

<31:2> MASTER READ ADDRESS 31 through 2 are write/read bits used to specify the initial DMA address during block transfer operations. Reading this register returns the last address accessed during a DMA operation.

<1:0> These bits are not used since the DMA operations execute longword (32-bit) transfers.

MASTER READ TRANSFER COUNT REGISTER

The Master Read Transfer Count Register is a write/read register located at an offset of 30 hex from the selected base address #1 and is loaded with the number of bytes to transfer from PCI memory to the 2915 during a DMA operation. This counter is decremented by four for every PCI bus cycle executed, since the 2915 transfers only 32-bit data words during DMA. When this counter is decremented to zero, the DMA operation terminates and an interrupt may be optionally generated to the PCI bus. If a CAMAC block transfer operation is terminated due to an error, this register may be read to determine the number of data bytes not transferred to the 2915 for the requested operation. The following diagram shows the bit pattern for the Master Read Transfer Count Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	MRTC 25	MRTC 24	MRTC 23	MRTC 22	MRTC 21	MRTC 20	MRTC 19	MRTC 18	MRTC 17	MRTC 16
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
MRTC 15	MRTC 14	MRTC 13	MRTC 12	MRTC 11	MRTC 10	MRTC 09	MRTC 08	MRTC 07	MRTC 06	MRTC 05	MRTC 04	MRTC 03	MRTC 02	MRTC 01	MRTC 00

<31:26> These bits are not used and read as zeros.

<25:00> MASTER READ TRANSFER COUNT 25 through 0 are write/read bits used to specify the maximum number of bytes to transfer from PCI memory to the 2915 during DMA operations.

MAILBOX EMPTY/FULL STATUS REGISTER

The register is not used by the 2915 but is shown here for completeness.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMB 4D	IMB 4C	IMB 4B	IMB 4A	IMB 3D	IMB 3C	IMB 3B	IMB 3A	IMB 2D	IMB 2C	IMB 2B	IMB 2A	IMB 1D	IMB 1C	IMB 1B	IMB 1A
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
OMB 4D	OMB 4C	OMB 4B	OMB 4A	OMB 3D	OMB 3C	OMB 3B	OMB 3A	OMB 2D	OMB 2C	OMB 2B	OMB 2A	OMB 1D	OMB 1C	OMB 1B	OMB 1A

PCI INTERFACE INTERRUPT CONTROL/STATUS REGISTER

The PCI Interface Interrupt Control/Status Register of the PCI Interface Operational Registers is located at an offset of 38 hex from the selected base address #1 and used to monitor and control interrupts generated by the PCI interface chip. Before an interrupt is sourced from the interface chip to the PCI bus, it must be enabled with the PCI INTERRUPT ENABLE bit in the Control/Status Register of the Parallel Bus Operational Registers.

The following shows the bit pattern for the Interrupt Control/Status Register of the PCI Interface Operational Registers.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	INT REQ	0	TAR ABT	MAS ABT	RTC	WTC	IMB SRC	OMB SRC
15	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**
RTC IE	WTC IE	0	IMB IE	IMB EMB	IMB EMA	IMB EBB	IMB EBA	0	0	0	OMB IE	OMB EMB	OMB EMA	OMB EBB	OMB EBA

<31:24> These 8 bits control endian conversion and must be set to zero for the 2915 to operate properly.

<23> INTERRUPT REQUEST is a read-only bit that is set when the PCI interface chip is requesting service. This bit reflects the interrupt output of the PCI chip and does not indicate that the PCI bus interrupt is asserted. If the PCI Interrupt Enable bit of the Interrupt Control/Status Register of the Parallel Bus Operational Registers is set to a one, this bit indicates that a PCI bus interrupt is requested.

<22> This bit is not used and read as a zero.

<21> TARGET ABORT is a read/write-to-clear bit that indicates when the 2915 executes a bus master transfer and the addressed target aborts the transfer. An interrupt source is generated when this bit is set. This bit is cleared by writing a one to this bit position.

<20> MASTER ABORT is a read/write-to-clear bit that indicates when the 2915 executes a bus master operation and the addressed target does not respond. An interrupt source is generated when this bit is set. A write operation to this register with this bit set to a one clears the bit.

<19> READ TRANSFER COMPLETE is a read/write-to-clear bit that is set when the Master Read Transfer Count Register is decremented to zero.

<18> WRITE TRANSFER COMPLETE is a read/write-to-clear bit that is set when the Master Write Transfer Count Register is decremented to zero.

<17> INCOMING MAILBOX INTERRUPT SOURCE is a read/write-to-clear bit that is set to a one when the mailbox selected by bits 12 through 8 of this register are written. Since the 2915 does not use the mailbox registers, this bit should not be set.

<16> OUTGOING MAILBOX INTERRUPT SOURCE is a read/write-to-clear bit that is set to a one when the mailbox selected by bits 4 through 0 of this register is read. Since the 2915 does not use the mailbox registers, this bit should not be set.

<15> READ TRANSFER COMPLETE INTERRUPT ENABLE is a write/read bit used to enable and disable the generation of an interrupt when the read transfer count is exhausted.

<14> WRITE TRANSFER COMPLETE INTERRUPT ENABLE is a write/read bit used to enable and disable the generation of an interrupt when the write transfer count is exhausted.

<13> This bit is not used and read as a zero.

<12> INCOMING MAILBOX INTERRUPT ENABLE is a write/read bit that is used to enable/disable the generation of an interrupt source when a preselected incoming mailbox

register is written. Bits 11 through 8 of this register select which mailbox register write operation will generate the interrupt source.

<11:10> INCOMING INTERRUPT SELECT bits are used to select which incoming mailbox write operations are to generate an interrupt source. Since the 2915 does not used the mailbox registers, these bits should be set to zero.

<9:8> INCOMING MAILBOX BYTE INTERRUPT SELECT bits are used to select which byte of the mailbox, selected by bits 11 and 10 of this register, is actually to cause the interrupt source. Since the 2915 does not used the mailbox registers, these bits should be set to zero.

<7:5> These bits are not used and read as zeros.

<4> OUTGOING MAILBOX INTERRUPT ENABLE is a write/read bit that is used to enable/disable the generation of an interrupt when a preselected outgoing mailbox register is written. Since the 2915 does not used the mailbox registers, these bits should be set to zero.

<3:2> OUTGOING INTERRUPT SELECT bits are used to select which outgoing mailbox write operations are to generate an interrupt source. Since the 2915 does not use the mailbox registers, these bits should be set to zero.

<1:0> OUTGOING MAILBOX BYTE INTERRUPT SELECT bits are used to select which byte of the mailbox, selected by bits 3 and 2 of this register, is actually to cause the interrupt source. Since the 2915 does not use the mailbox registers, these bits should be set to zero.

BUS MASTER CONTROL/STATUS REGISTER

The Bus Master Control/Status Register is used to monitor/control bus master operations and to check status of the two PCI interface chip FIFOs. The following diagram shows the bit pattern for the Bus Master Control/Status Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	MBX RST	INF RST	OTF RST	AON RST	0	0	0	0	0	0	0	0

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0	RDT ENA	0	0	0	WTT ENA	0	0	ITC ZERO	OTC ZERO	IFE MT	IFI 4+	IFI FUL	OTF MT	OTF 4+	OTF FUL

The bits shown as zeros in the preceding bit layout must be set to zeros when writing to the Bus Master Control/Status Register. The bits that have non-zero indications are the only useful bits in this register. The remaining bits are reserved for diagnostic purposes. Even though some of these bits are shown as zeros, they may eventually be read as ones. Note that the Inbound FIFO refers to the internal PCI controller interface chip's FIFO for executing CAMAC read operations and the Outbound FIFO refers to CAMAC write operations.

<31:28> These bits are not used but must be written to zero.

<27> MAILBOX FLAG RESET is a write-only bit which resets all of the mailbox status flags.

<26> INBOUND FIFO RESET is a write-only bit which clears the inbound FIFO, Inbound FIFO +4 bit, Inbound FIFO Full and sets the Inbound FIFO Empty flag.

<25> OUTBOUND FIFO RESET is a write-only bit which clears the outbound FIFO, Outbound FIFO +4 bit, Outbound FIFO Full and sets the Outbound FIFO Empty bit.

<24> ADD-ON RESET is a write-only bit which, when set to a one, resets all of the Parallel Bus Operational Registers.

<23:15> These bits are not used but must be written to zero.

<14> READ TRANSFER ENABLE (RDT ENA) is a write/read bit used to enable DMA operations from PCI memory to the 2915 for CAMAC write operations. Setting this bit to a zero before the transfer count expires suspends the active transfer.

<13:11> These bits are not used but must be written to zero.

<10> WRITE TRANSFER ENABLE (WTT ENA) is a write/read bit used to enable DMA operations from the 2915 to PCI memory for CAMAC read operations. Setting this bit to a zero before the transfer count expires suspends the active transfer.

<09:08> These bits are not used but must be written to zero.

<07> INBOUND TERMINAL COUNT ZERO (ITC ZERO) is a read-only bit that is set to a one indicating that the Master Write Transfer Count is zero.

<06> OUTBOUND TERMINAL COUNT ZERO (OTC ZERO) is a read-only bit that is set to a one indicating that the Master Read Transfer Count is zero.

<5> INBOUND FIFO EMPTY is a read-only bit which is set when the inbound FIFO contains no data.

<4> INBOUND FIFO +4 is a read-only bit and is set to a one as long as there are at least 4 longwords of data in the inbound FIFO.

<3> INBOUND FIFO FULL is a read-only bit that is set when the incoming FIFO is full.

<2> OUTBOUND FIFO EMPTY is a read-only bit which is set when the outbound FIFO contains no data.

<1> OUTBOUND FIFO +4 SPACES is a read-only bit and is set to a one as long as there are at least 4 empty longwords in the outbound FIFO.

<0> OUTBOUND FIFO FULL is a read-only bit that is set when the outbound FIFO is full.

PARALLEL BUS OPERATIONAL REGISTERS

The following section describes the registers used to control and monitor operations directed toward the Parallel Bus. The base address of these registers is set dynamically when the computer is power-up and the BIOS routines are executed. The PCI BIOS determines the resources required by each add-in device and allocates memory and I/O resources accordingly. After the setup has been completed, the Base Address Register #2 from the PCI Interface Registers may be read to determine the base address allocated to the 2915 Parallel Bus Operational Registers by the PCI BIOS.

The following chart shows the various registers used to control/monitor the Parallel Bus operations.

Address	Register Description	Mnemonic
Base + 00	Control/Status Register	CSR
Base + 04	CAMAC Crate/Command Register	CNAF
Base + 08	Transfer Count Register	TCR
Base + 0C	Service Request Register	SRR

Appendix A contains a composite register layout chart for the 2915.

CONTROL/STATUS REGISTER

The Control/Status Register (CSR) is a write/read register located at an offset of zero from the selected base address. This register is used to control and monitor various operations occurring within the 2915 and on the Parallel Bus. Since this register contains write-only, read-only, and write/read bits, two bit patterns are shown. Those bits that are shown as zero for the write layout must be set to zero when writing to this register.

Control/Status Register (CSR)

Write Operations:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	160
0	0	0	RST INFC	0	0	0	0	0	0	0	0	0	0	0	0
15	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**
0	0	WORD SIZE	ABT DIS	PCI IRQ	PCI IENA	0	RFS IENA	0	DONE IENA	CLR PCII	CLR DNI	MD 4	MD 2	MD 1	GO

Read Operations:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERR	0	0	0	0	0	0	0	0	0	0	BUF FULL	PBUS TMO	NAF TMO	NOX	NOQ
15	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**
0	0	WORD SIZE	ABT DIS	PCI IRQ	PCI IENA	RFS	RFS IENA	DONE	DONE IENA	0	0	MD 4	MD 2	MD 1	0

<31> ERROR is a read-only bit that is set when an error occurs during a Parallel Bus operation. This error may result from either a CAMAC Q or X response of non-zero, a Parallel Bus Timeout during a CAMAC operation or a NAF transaction.

The following equation describes error.

```
ERROR = PBUS TMO
        + NAF TMO
        + !MD4 & !MD2 & MD1 & NO-Q                ( Q-Stop Block)
        + !MD4 & MD2 & !MD1 & NO-X & !ABTDIS      ( Q-Ignore Block)
        + !MD4 & !MD2 & MD1 & NO-X & !ABTDIS      ( Q-Stop Block)
        + !MD4 & MD2 & MD1 & NO-X & !ABTDIS       ( Q-Repeat Block)
        + !MD4 & MD2 & MD1 & QTMO                 ( Q-Repeat Block)
        + MD4 & !MD2 & !MD1 & N>23                ( Q-Scan Block)
```

PBUS TMO	- Parallel Bus Timeout during CAMAC Operation (Bit 18 of the CSR)
NAF TMO	- CAMAC NAF Transfer Timeout (Bit 18 of the CSR)
MD4	- Mode 4 (Bit 3 of the CSR)
MD2	- Mode 2 (Bit 2 of the CSR)
MD1	- Mode 1 (Bit 1 of the CSR)
ABTDIS	- Abort on NO-X Disable
QTMO	- Q-Repeat Timeout (Q-Repeat operation did not receive a CAMAC Q-response of 1 within the 60 millisecond timeout period)
N>23	- N Greater than 23 (Q-Scan operation caused Station Number to increment beyond slot 23.

<30:29> These bits are not used and read as zeros.

<28> RESET INTERFACE (RST INFC) is a write-only bit which resets the 2915 Parallel Bus circuitry to a power-up state. Setting this bit to a one causes the reset operation but does not affect any of the PCI Configuration register setup. This bit is not latched and read as a zero.

<27:20> These bits are not used and read as zeros.

<19> PARALLEL BUS TIMEOUT (PBUS TMO) is a read-only bit which is set when a requested CAMAC operation causes the Parallel Bus to timeout. The timeout circuit is started when the 2915 requests to transfer a byte of data to/from the 3922. If the 2915 does not receive a response in 200 milliseconds, a Parallel Bus Timeout is signaled and the operation terminates.

<18> CAMAC NAF TRANSFER TIMEOUT (NAF TMO) is a read-only bit that is set when the 2915 attempts an access to a 3922 CAMAC Command Register (NAF) and does not receive a response from the addressed chassis. After a NAF TMO is generated, the operation is terminated and the DONE bit is asserted. This bit is reset when a CAMAC operation is requested or a NAF Transfer is initiated.

<17> NO-X is a read-only bit which is set when an addressed CAMAC operation results with a CAMAC X-response of zero.

<16> NO-Q is a read-only bit which is set when an addressed CAMAC operation results with a CAMAC Q-response of zero.

<15:14> These bits are not used and read as zeros.

<13> WORD SIZE is a write/read bit used to define the size of the CAMAC data words used during addressed CAMAC operations. Setting this bit to a zero results in 24-bit CAMAC data words and a one selects 16-bit operations. When 24-bit operations are selected, each

CAMAC data word is contained in each 32-bit PCI data word. When 16-bit transfers are selected, two 16-bit CAMAC data words are contained in each PCI data word.

<12> ABORT DISABLE (ABT DIS) is a write/read bit used to specify the action to be taken when a CAMAC NO-X response is received during a block transfer operation. This bit is set to a one to prevent a CAMAC block transfer operation from terminating due to a CAMAC X-response of zero. If the ABT DIS bit is set to a zero and an X-response of zero is received, the block terminates and ERROR is asserted. This bit is not used during Scan operations since X-responses of zero may normally occur.

<11> PCI INTERRUPT REQUEST (PCI IRQ) is a read-only bit that is asserted when the PCI interface is requesting service. For the PCI interface to generate an interrupt to the PCI bus, the PCI INTERRUPT ENABLE (bit 10 in the CSR) must be set to a one. This bit is cleared as long as the PCI interface has all its interrupt sources cleared. Please refer to the PCI Interface Operational Registers section of this manual for additional information on the sources of these interrupts.

<10> PCI INTERFACE INTERRUPT ENABLE (PCI IENA) is a write/read bit used to enable/disable the PCI interface from generating an interrupt to the PCI bus. Setting this bit to a one enables the interrupt and a zero disables the interrupt.

<09> REQUEST FOR SERVICE (RFS) is a read-only bit that is set when a CAMAC Look-At-Me (LAM) is pending in any 3922. This signal is wire-ORed from all 3922s on the Parallel Bus. In order to determine which 3922(s) is requesting service, a Parallel Poll operation must be executed to find the 3922(s) asserting the request(s). Please refer to the Parallel Poll section of this manual for additional information.

<08> REQUEST FOR SERVICE INTERRUPT ENABLE is a write/read bit used to enable/disable the 2915 from generating a PCI interrupt when the REQUEST FOR SERVICE (RFS) bit is asserted. Setting this bit to a one enables the interrupt and a zero disables the interrupt generation.

<07> DONE is a read-only bit used to indicate when the 2915 is idle. This bit is set on power-up and also after a requested operation is complete. When the GO bit (Bit 0 in the CSR) is set, the DONE bit is negated while the operation is being performed.

<06> DONE INTERRUPT ENABLE (DONE IENA) is a write/read bit used to enable/disable the 2915 from generating a PCI interrupt once the DONE bit (Bit 7 of the CSR) is asserted. Setting this bit to a one enables the interrupt source and disabled by setting the bit to a zero. If the DONE bit is set and the DONE IENA is then set, an interrupt will not be generated until the GO bit is set and an operation completes.

<05> CLEAR PCI INTERFACE INTERRUPT SOURCE is a write-only bit used to clear the PCI interface chips assertion of an interrupt request to the PCI bus. Setting this bit to a one when writing the CSR clears the interrupt request, but does not clear the interrupt source. Please refer to the PCI Interface Operational Registers section of this manual for additional interrupt information.

<04> CLEAR DONE INTERRUPT SOURCE is a write-only bit used to clear the DONE interrupt source on the 2915 once an interrupt has been generated. Setting this bit to a one when writing to the CSR clears the DONE interrupt source.

<03:01> MODE SELECT4 ,2 and 1 are write/read bits used to define the operation the 2915 is to perform once the GO bit is set. The binary combinations of these bits specify the actual

modes. Please refer to the Operating Modes section of this manual for additional information on the various operating modes. The following chart shows the operations executed based on the binary combination of these bits.

Mode 4	Mode 2	Mode 1	Selected Operation
0	0	0	Single Transfer
0	0	1	Q-Stop Block Transfer
0	1	0	Q-Ignore Block Transfer
0	1	1	Q-Repeat Block Transfer
1	0	0	Q-Scan Block Transfer
1	0	1	Parallel Poll Operation
1	1	0	CAMAC NAF Read Operation
1	1	1	CAMAC NAF Write Operation

<00> GO is a write-only bit used to initiate an operation as specified by the Mode Select bits 3 through 1 of the CSR. Once the GO bit is set to a one, the DONE bit in the CSR is negated. Once the requested operation is complete, the DONE bit is re-asserted. The 2915 is then ready to execute additional operations.

CAMAC CRATE/COMMAND REGISTER

The CAMAC Crate/Command Register (CNAF) is a write/read register located at an offset of 4 from the selected base address. This register is used for communicating CAMAC Station Number (N), Subaddress (A), and Function Code (F) information to/from an addressed CAMAC chassis. This register also contains the Crate Address © to be accessed during the operation.

The Crate Address bits are used to define the chassis to be accessed during a CAMAC or NAF Parallel Bus transaction. The three bits allow for the eight possible Crate Addresses ranging from Crate Address 0 through Crate Address 7.

The CAMAC NAF portion of this register is used to specify the CAMAC Station Number, Subaddress and Function Code used during a CAMAC transfer mode. Each 3922 contains its own individual NAF register that is actually used during a CAMAC operation. Therefore, the NAF register on a particular 3922 must be loaded prior to executing the CAMAC transfer. This is done automatically by the 2915 when a CAMAC transfer operation is selected and the GO bit of the CSR is set. Before the CAMAC transfer is executed, the 2915 transfers two bytes of CAMAC NAF data to the addressed 3922 before switching to the CAMAC data transfer portion of the cycle. The NAF data must be preloaded in the CNAF register before the GO bit is set.

Two diagnostic mode selections are provided to access the individual NAF registers on each 3922. A Mode 6 operation is used to read a 3922 NAF register and a Mode 7 is used to write to a 3922 NAF register. For Mode 6 operations, the data written to the 3922 NAF register must be loaded in the 2915 CNAF register prior to executing the operation. After a Mode 7 operation is executed, the NAF register contents on the 3922 are transferred into the 2915 CNAF register. This provides a mechanism for determining NAF data integrity on individual 3922s.

The following diagram shows the bit pattern for the CAMAC Crate/Command Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	C4	C2	C1

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0	0	N 16	N 8	N 4	N 2	N 1	A 8	A 4	A 2	A 1	F 16	F 8	F 4	F 2	F 1

<31:19> These bits are not used and read as zeros.

<18:16> CRATE ADDRESS 4, 2 and 1 are write/read bits used to specify the Crate Address to be accessed for subsequent CAMAC transfers or NAF transfer operations. These three bits allow selection of controllers 0 through 7.

<15:14> These bits are not used and read as zeros.

<13:09> STATION NUMBER 16 through 1 (N16 -N1) are write/read bits used to specify the CAMAC Station Number to be used during an addressed CAMAC operation. These bits allow for selections of 0 through 31. Station Numbers 0, 24 through 29 and 31 are not used. Station Number 30 is a pseudo-address used to access the internal registers on the 3922.

<08:05> SUBADDRESS 8 through 1 (A8 - A1) are write/read bits used to specify the CAMAC Subaddress to be used during an addressed CAMAC operation. These bits allow for a selection of Subaddresses in the range of 0 through 15.

<04:00> FUNCTION CODE 16 though 1 (F16 - F1) are write/read bits used to specify the CAMAC Function Code to be used during the addressed CAMAC operation. These bits allow for a Function Code selection in the range of 0 through 31. The binary combination of the F16 and F8 bits determine the type of CAMAC transfer as shown in the following chart.

F16	F8	Operation Type
0	0	CAMAC Read
0	1	CAMAC Control
1	0	CAMAC Write
1	1	CAMAC Control

CAMAC Write operations are data transfers from the 2915 to the addressed CAMAC module
CAMAC Read operations are data transfers from the addressed CAMAC module to the 2915.
CAMAC Control operations are dataless operations.

TRANSFER COUNT REGISTER

The Transfer Count Register (TCR) is a write/read register located at an offset of 8 from the selected base address. This register is used to specify the maximum number of transfers that are to occur when a block transfer mode of operation is executed. The format of this data must be in two's complement and specifies the number of CAMAC operations to be executed, regardless of the CAMAC data word size selected.

Before block transfer read operations are initiated, the PCI Interface Master Write Transfer Count Register must be loaded with the byte count for the operation and the TCR must be loaded with the CAMAC data word count. When the block operation is executed, the 2915 obtains CAMAC read data until the TCR is incremented to zero. The PCI Interface continues transferring data to the PCI bus as long as CAMAC read data is available and the Master Write Transfer Count Register has not expired.

Similarly, before block transfer write operations are initiated, the PCI Interface Master Read Transfer Count Register must be loaded with the byte count for the operation and the TCR must be loaded with the CAMAC data word count. When the block operation is executed, the 2915 obtains CAMAC write data until the Master Transfer Count Register is decremented to zero. The 2915 continues transferring data to the CAMAC chassis until the TCR is incremented to zero.

If an error occurs during a block transfer mode of operation, the Transfer Count Register in conjunction with either the Master Read Transfer Count Register or the Master Write Transfer Count Register can be used to determine the actual number of transfers successfully executed.

The following diagram shows the bit pattern for the Transfer Count Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	TCR 23	TCR 22	TCR 21	TCR 20	TCR 19	TCR 18	TCR 17	TCR 16
15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
TCR 15	TCR 14	TCR 13	TCR 12	TCR 11	TCR 10	TCR 09	TCR 08	TCR 07	TCR 06	TCR 05	TCR 04	TCR 03	TCR 02	TCR 01	TCR 00

<31:24> These bits are not used and read as zeros.

<23:00> TRANSFER COUNT 23 through 0 are write/read bits used to specify the two's complement of the maximum number of transfers that are to occur during a block transfer mode of operation. If an error is encountered, this register may be read and used in conjunction with the Master Write/Read Transfer Count Register of the PCI Interface Registers to determine the actual number of valid transfers executed.

SERVICE REQUEST REGISTER

The Service Request Register (SRR) is a read-only register located at an offset of 0C hex from the selected base address. This register is used to hold the data obtained from executing a Parallel Poll operation. The Parallel Poll operation returns status from each 3922 regarding its Request For Service. Each Crate Address has a bit position in the SRR. Any bit read back as a one indicates the corresponding chassis has a Request For Service pending.

A Parallel Poll operation is initiated by setting the mode bits in the CSR to 5 and setting the 'GO' bit. Once the Parallel Poll operation in done, the SRR may then be read. When the GO bit in the CSR is set, the DONE bit is negated until the Parallel Poll is complete.

The following diagram shows the bit pattern for the Service Request Register.

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
0	0	0	0	0	0	0	0	C 7	C 6	C 5	C 4	C 3	C 2	C 1	C 0

<31:08> These bits are not used and read as zeros.

<07:00> CRATE ADDRESS 7 through 0 are read-only bits which reflect the chassis asserting a Request For Service after the 2915 executes a Parallel Poll operation.

PROGRAMMED I/O TRANSFERS

The 2915 provides for programmed control modes of operation. In this mode, all 2915 initialization is done by programmed I/O and all data transfers to/from the CAMAC chassis are also moved using programmed control. For faster data transfers, the Direct Memory Address (DMA) mode of operation should be used. Refer to the DMA Transfers section of this manual for additional information.

During programmed transfer operations, all data transferred to/from the 2915 is under control of the host processor. For CAMAC write operations, the host processor must supply the CAMAC write data by writing the data to the Data FIFO Register in the PCI Interface. Similarly, when executing CAMAC read operations, the host processor must retrieve the data obtained from the CAMAC read in the Data FIFO Register in the PCI Interface.

Either the Single Transfer Mode (Mode 0) or one of the Block Transfer Modes (Modes 1 through 4) may be executed using programmed transfers. The following sections explain the procedure required to execute Single Transfer Programmed Transfers and Block Transfer Programmed Transfers.

SINGLE PROGRAMMED I/O TRANSFERS

A Single Transfer Mode of operation simply transfers one CAMAC data word to/from a CAMAC crate. This mode is selected by setting the M4, M2 and M1 bits in the Control/Status Registers (CSR) to a zero when writing the GO bit. After the GO bit is set for a CAMAC write operation, the CAMAC write data must then be loaded into the PCI Interface FIFO Data Register. For CAMAC read operations, the CAMAC read data can be read from the PCI Interface FIFO Data Register.

Care must be taken when writing data to the FIFO or reading data from the FIFO. Before an access attempt is made, the PCI Interface Bus Master Control/Status Register should be read to determine the amount of data contained in the FIFOs. When writing to the FIFO, ensure that there is room available for the FIFO to accept the write data. For FIFO read operations, ensure that the FIFO contains data to be read. Failure to check the FIFO flags before accessing the FIFO Data Register could result in the PCI bus timing out due to a large number of PCI retry and disconnects. It should also be noted that a CAMAC read that results in an error does not deposit any read data into the FIFO.

The following procedures should be followed for executing Single Transfer Programmed Transfers.

Programmed Transfer Read Operation (F16 = 0 ; F8 = 0)

1.) Load the CAMAC Crate Address and the CAMAC read command to be executed in the CAMAC Crate/Command Register (CNAF).

2.) Load the Control/Status Register (CSR) with M4, M2 and M1 set to 0 and GO set to a 1.

3.) Wait for the DONE bit in the CSR to be asserted.

4.) Check the Inbound FIFO Flag in the Bus Master CSR to see if CAMAC read data is available for reading.

5.) If CAMAC read data is available, retrieve it from the Data FIFO Register.

6.) Check the ERR, No-X and No-Q bits in the CSR to verify data validity.

Programmed Transfer Write Operation (F16 = 1 ; F8 = 0)

1.) Load the CAMAC Crate Address and the CAMAC write command to be executed in the CAMAC Crate/Command Register (CNAF).

2.) Load the Control/Status Register (CSR) with M4, M2 and M1 set to 0 and GO set to a 1.

3.) Check the Outbound FIFO Flag in the Bus Master CSR to see if there is room in the FIFO for the CAMAC write data.

4.) Write the CAMAC write data into the FIFO Data Register.

5.) Wait for the DONE bit in the CSR to be asserted.

6.) Check the ERR, No-X and No-Q bits in the CSR to verify the addressed module accepted the data.

Programmed Transfer Control Operation (F8 = 1)

1.) Load the CAMAC Crate Address and the CAMAC control command to be executed in the CAMAC Crate/Command Register (CNAF).

2.) Load the Control/Status Register (CSR) with M4, M2 and M1 set to 0 and GO set to a 1.

3.) Wait for the DONE bit in the CSR to be asserted.

4.) Check the ERR, No-X and No-Q bits in the CSR to verify transfer integrity.

Note: When executing transfers with the DONE interrupt enabled, the step that polls for the DONE bit may be omitted since the assertion of the DONE bit generates an interrupt.

BLOCK TRANSFER OPERATIONS

The 2915 provides four modes of transferring a block of data to/from a CAMAC crate. These modes include Q-Stop, Q-Ignore, Q-Repeat and Q-Scan. The block transfer data may be transferred to/from the 2915 using programmed transfers or direct memory access (DMA). For programmed transfers, all data transferred to/from the 2915 is under control of the host processor. A faster and more efficient mechanism to transfer CAMAC write/read data is using Direct Memory Access (DMA). Once DMA operations are initialized, all data transfers to/from the 2915 during a block transfer operation are executed

autonomously. The host processor merely initiates the operation and the 2915 moves the data to and from the host memory.

During CAMAC block transfer read operations, the initial CAMAC read operation is executed. After the first command is complete, the data is checked for validity. Data validity depends on such parameters as the Transfer Mode selected and the CAMAC Q- and X-responses. Refer to the individual block transfer mode types for additional information. If valid data is obtained, the 2915 transfers the data to the Data FIFO Register. This data may then be read by the host or DMA-ed to the host memory.

If the Transfer Count has not been exhausted, the 2915 requests additional CAMAC read operations until the counter is exhausted. Subsequent read data words are then written into the Data FIFO Register. The sequence continues until the transfer count is exhausted or an error is encountered. In either case, the DONE bit in the CSR is set to indicate the operation is complete. If an error caused the CAMAC block operation to prematurely terminate, the Transfer Count Register (TCR) can be read to determine the number of data words not transferred. The TCR is initially loaded with the two's complement of the maximum number of transfers to occur for the block. This value is the number of CAMAC cycles to execute, regardless of the CAMAC data word size selected. As each request is made over the Parallel Bus for a CAMAC read operation, the TCR is incremented. Since the counter is incremented for each read request, regardless of data integrity, the resultant number in the TCR must be decremented by one. For example, if a value of 00FFFFFF hex is found in the TCR after an error occurs, the actual number of transfers not executed is 2.

During CAMAC block transfer write operations, the 2915 waits for the Data FIFO Register to contain the CAMAC write data. The data is written to the Data FIFO Register by either the host processor or by DMA. Once the write data is received, a CAMAC write operation is requested over the Parallel Bus. After the first command is complete, the transfer is checked for validity. Transfer validity depends on such parameters as the Transfer Mode selected and the CAMAC Q- and X-responses. Refer to the individual block transfer mode types for additional information. If a valid transfer was executed, the next write data is fetched from the Data FIFO Register and the operation continues.

If the Transfer Count has not been exhausted, the 2915 requests additional CAMAC write data words from the Data FIFO Register and executes the CAMAC operation until the counter is exhausted. The sequence continues until the transfer count is exhausted or an error is encountered. In either case, the DONE bit in the CSR is set to indicate the operation is complete. If an error caused the CAMAC block operation to prematurely terminate, the Transfer Count Register (TCR) can be read to determine the number of data words not transferred. The TCR is initially loaded with the two's complement of the maximum number of transfers to occur for the block. This value is the number of CAMAC cycles to execute, regardless of the CAMAC data word size selected. As each request is made over the Parallel Bus for a CAMAC write operation, the TCR is incremented. Since the counter is incremented for each write request, regardless of data integrity, the resultant number in the TCR must be decremented by one. For example, if a value of 00FFFFFF hex is found in the TCR after an error occurs, the actual number of transfers not executed is 2. The Control/Status Register of the 3922 must also be examined to see if a CAMAC write data word is present in the first stage of the double-buffer. If the Buffer Full bit in the CSR is set to a one, the number of words not transferred must be incremented by one. This is only used for CAMAC write operations.

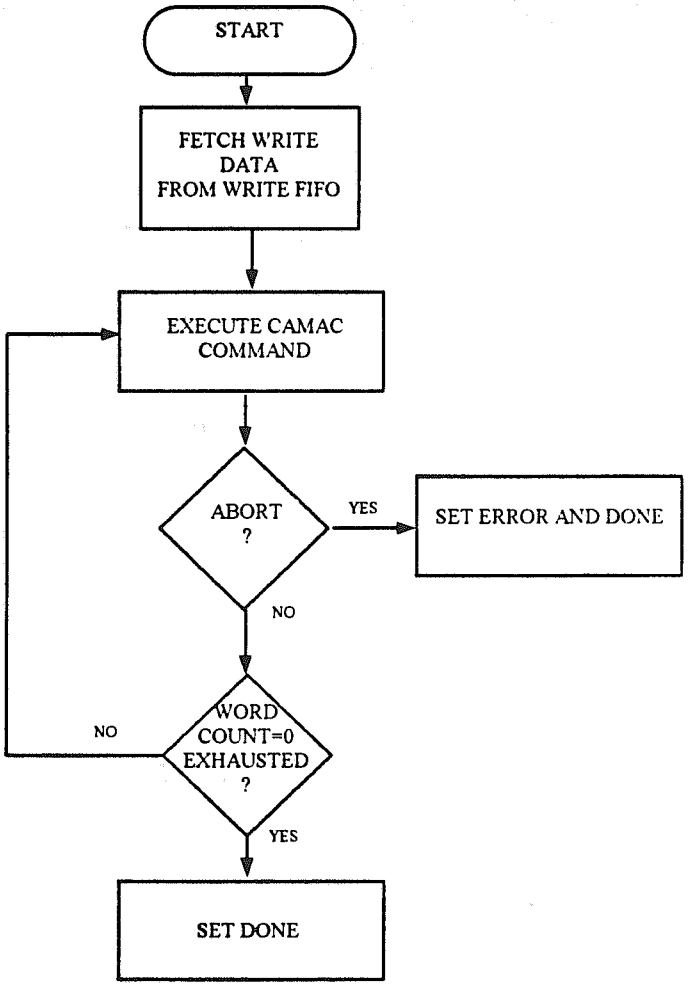
Q-STOP BLOCK TRANSFERS

The Q-Stop block transfer mode is selected by setting the M4 and M2 to zero and the M1 bit to a one when setting the GO bit in the CSR. During Q-Stop operations, the CAMAC command specified in the CAMAC Crate/Command Register (CNAF) is repeated until a Q-response of zero is received or the transfer count is exhausted. The block transfer also terminates in an error if the an X-response of zero is received with the abort enabled.

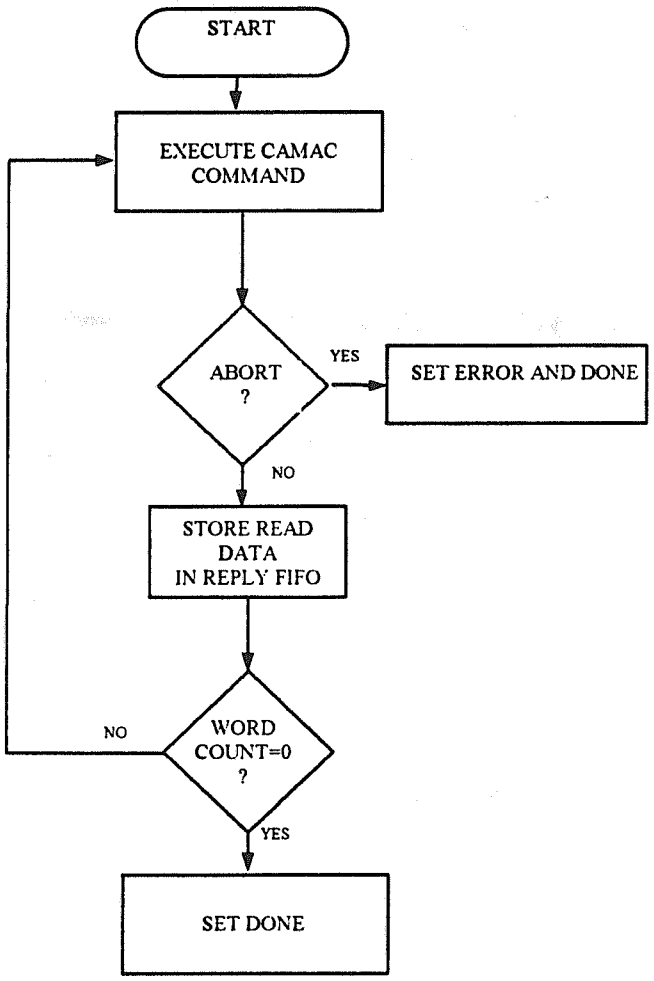
The following equation describes ERROR for the Q-Stop block transfer mode.

ERROR = NAF TMO
# PBUS TMO
# NO-Q
# NO-X * !AD

The following two diagrams are simplified flow diagrams illustrating Q-Stop block transfer write and read operations.



Q-IGNORE /Q-STOP WRITE COMMANDS



Q-IGNORE /Q-STOP READ COMMANDS

Q-IGNORE BLOCK TRANSFERS

The Q-Ignore block transfer mode is selected by setting the M4 and M1 bits to zero and the M2 bit to a one when setting the GO bit in the CSR. During Q-Ignore operations, the CAMAC command specified in the CAMAC Crate/Command Register (CNAF) is repeated until the transfer count is exhausted. The block transfer also terminates in an error if the an X-response of zero is received with the abort enabled. The following equation describes ERROR for the Q-Ignore block transfer mode.

ERROR = NAF TMO
# PBUS TMO
# NO-X * !AD

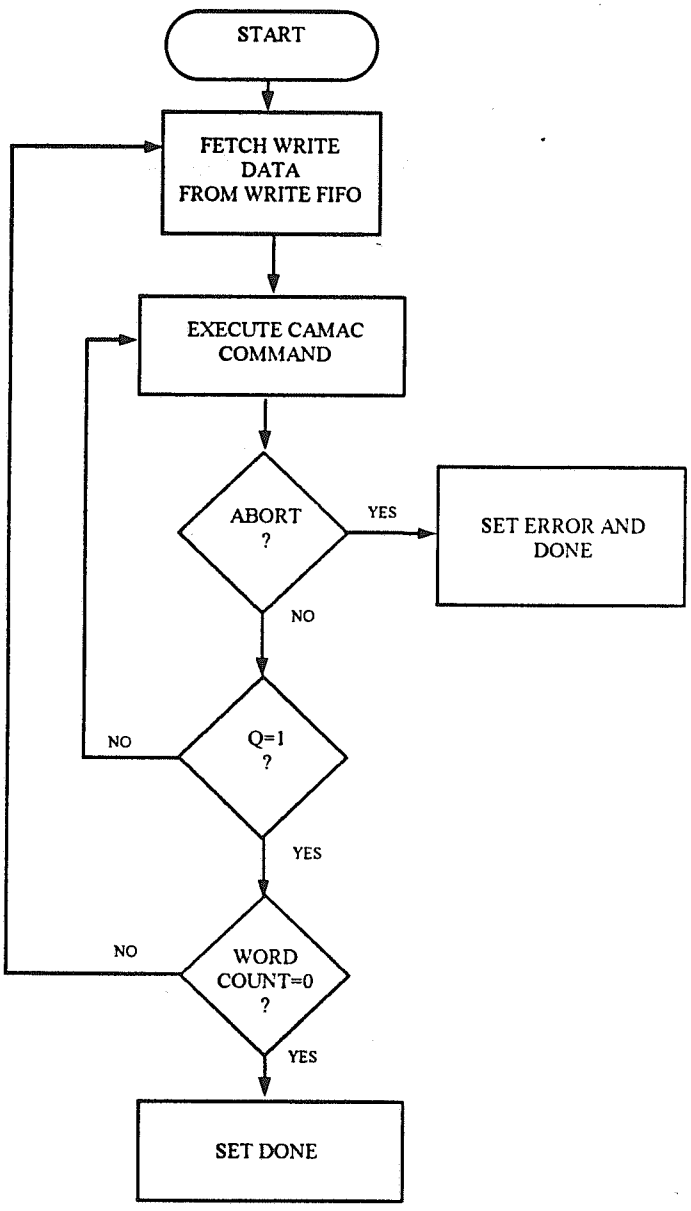
The two diagrams on page 21 are simplified flow diagrams illustrating Q-Ignore transfer write and read operations.

Q-REPEAT BLOCK TRANSFERS

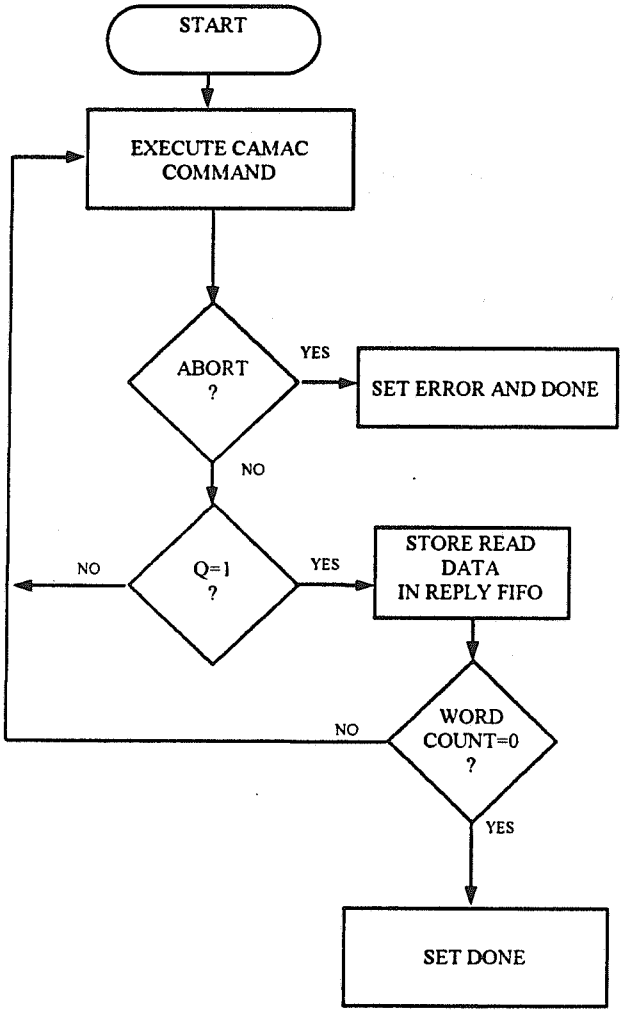
The Q-Repeat block transfer mode is selected by setting the M4 bit to a zero and the M2 and M1 bits to a one when setting the GO bit in the CSR. During Q-Repeat operations, the CAMAC command specified in the CAMAC Crate/Command Register (CNAF) is repeated for each data word until a CAMAC Q-response of one is received. A Q-response of one either causes new write data to be fetched or read data to be stored. The command is repeated for each data word until the transfer count is exhausted. If a Q-response of one is not received within 200 milliseconds, the ERROR bit is set and the block transfer terminates. The block transfer also terminates in an error if the an X-response of zero is received with the abort enabled. The following equation describes ERROR for the Q-Repeat block transfer mode.

ERROR = NAF TMO
# PBUS TMO
# Q-REPEAT TIMEOUT
# NO-X * !AD

The following diagrams are a flow diagrams for Q-Repeat block transfer write and read operations.



Q-REPEAT WRITE COMMANDS



Q -REPEAT READ COMMANDS

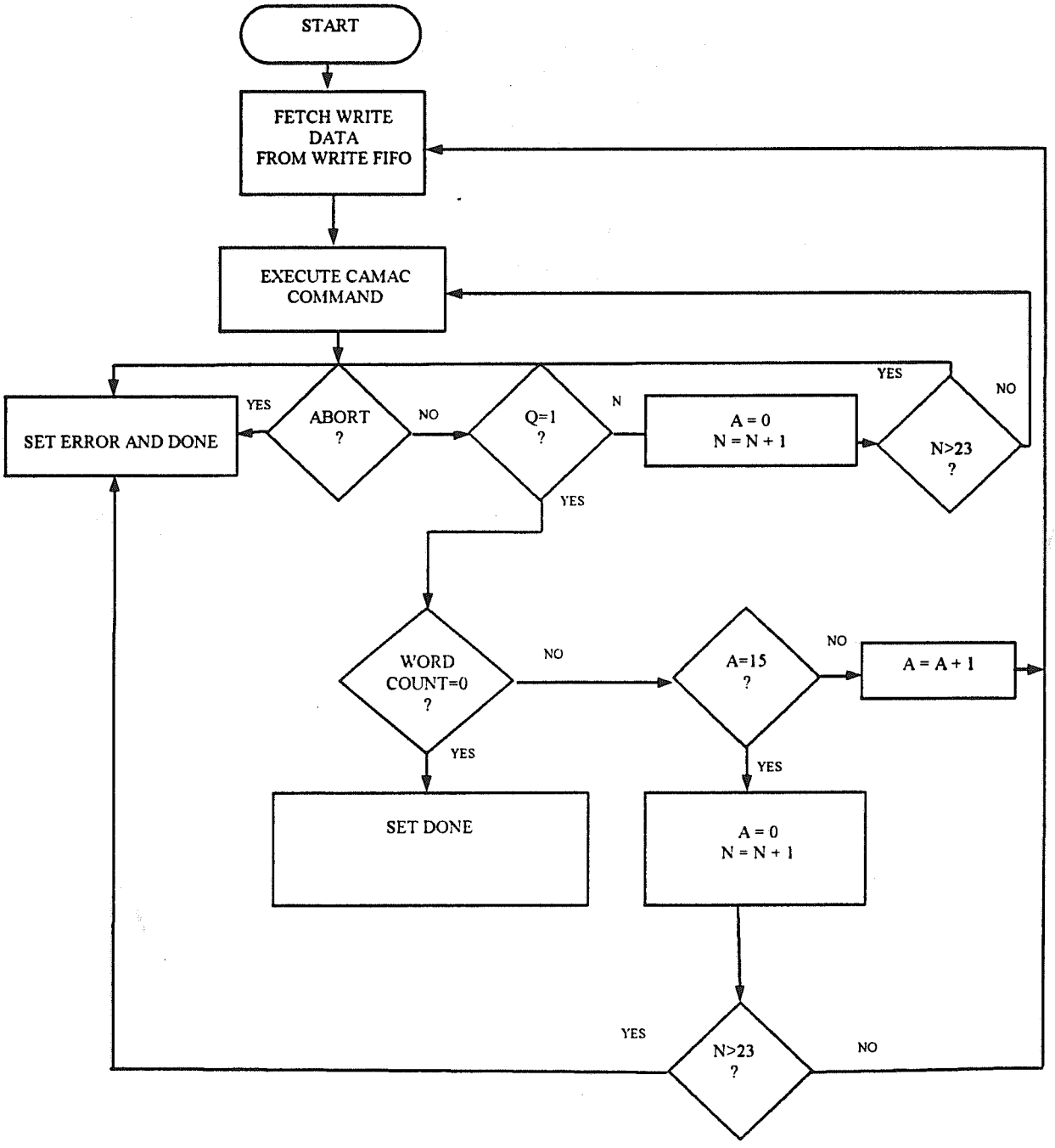
Q-SCAN BLOCK TRANSFERS

The Q-Scan block transfer mode is selected by setting the M4 bit to a one and the M2 and M1 bits to a zero when writing the GO bit in the CSR. During Q-Scan operations, the 2915 uses the Q-response from the previous operation to determine the station number (N) and subaddress (A) for the next operation. A Q-response of zero indicates that the last valid subaddress of the current station number has been accessed. The 2915 responds to a Q-response of zero by resetting the subaddress and incrementing the station number, and continuing the scan. A-Q-response of one indicates that the last command was executed to a valid subaddress. The 2915 responds to the Q-response of one by either storing the read data or fetching the next write data. After a Q-response of one is received, the CAMAC address is incremented as follows: the subaddress is incremented or if the subaddress was 15, it is reset to zero and the station number is incremented.

If, due to a programming error, the operation causes the station number to increment beyond station number 23, the block transfer is terminated and the ERROR bit is set. The following equation describes ERROR for the Q-Scan block transfer mode.

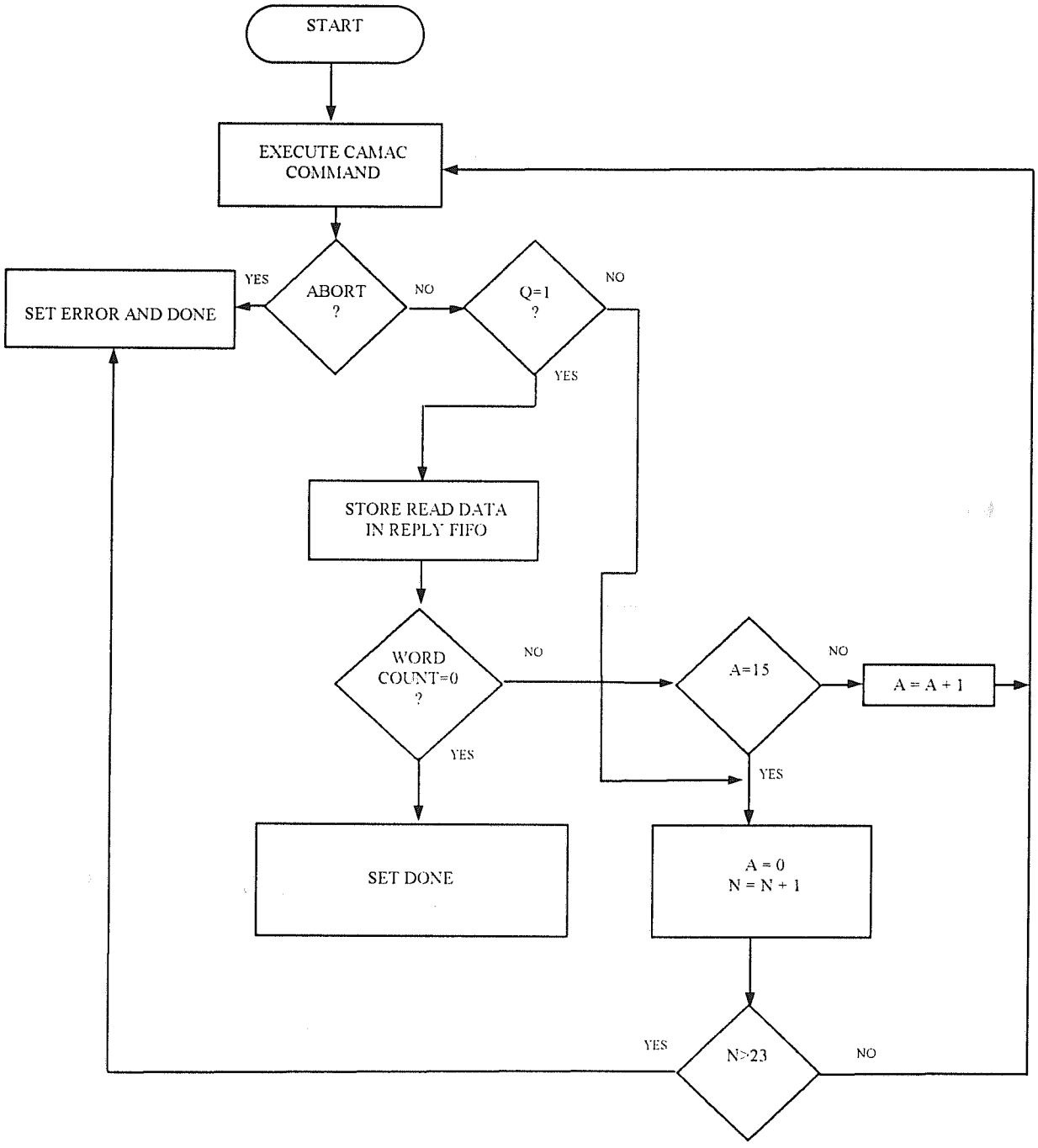
ERROR = NAFTMO
# PBUSTMO
# N>23

The following flow diagrams illustrate the Q-Scan write and read operations.



Q-SCAN WRITE OPERATIONS

The following flow diagrams illustrate the Q-Scan write and read operations.



Q-SCAN READ OPERATIONS

PROGRAMMED I/O BLOCK TRANSFERS

A Block Transfer Mode of operation transfers multiple CAMAC data words to/from a CAMAC crate. These modes provide a more efficient mechanism for transferring a block of data to/from a CAMAC module. This section describes the steps necessary to execute a block transfer operation using programmed I/O transfers to move the data to/from the interface. Another mechanism is also available for transferring data to/from the interface during block transfer operations. This mode is Direct Memory Access (DMA). Using DMA transfers, initial conditions are loaded by programmed I/O and then all subsequent data transfers to/from the interface are executed without host processor intervention. Please refer to the DMA Block Transfer section of this manual for additional information.

During programmed I/O block transfers, the host processor must transfer all data to/from the FIFO Data Register. Care must be taken when writing data to the FIFO or reading data from the FIFO. Before an access attempt is made, the PCI Interface Bus Master Control/Status Register should be read to determine the amount of data contained in the FIFOs. When writing to the FIFO, ensure that there is room available for the FIFO to accept the write data. For FIFO read operations, ensure that the FIFO contains data to be read. Failure to check the FIFO flags before accessing the FIFO Data Register could result in the PCI bus timing out due to a large number of PCI retry and disconnects. It should also be noted that a CAMAC read that results in an error does not deposit any read data into the FIFO.

Programmed Block Transfer Read Operation (F16 = 0 ; F8 = 0)

1.) Load the CAMAC Crate Address and the CAMAC read command to be executed in the CAMAC Crate/Command Register (CNAF).

2.) Load the Transfer Count Register with the two's complement of the number of transfers to execute.

3.) Load the Control/Status Register (CSR) with M4, M2 and M1 set appropriately and GO set to a 1.

4.) Check the Inbound FIFO Flag in the Bus Master CSR to see is CAMAC read data is available for reading.

5.) If CAMAC read data is available, retrieve it from the FIFO Data Register.

6.) Check the DONE bit in the CSR to see if the interface has completed the CAMAC accesses.

7.) If DONE is not set, jump back to step 4.

8.) If DONE is set, check the FIFO flag to ensure all read data has been read from the FIFO Data Register.

9.) Check the ERR, No-X and No-Q bits in the CSR to verify data validity.

Programmed Block Transfer Write Operation (F16 = 1 ; F8 = 0)

1.) Load the CAMAC Crate Address and the CAMAC read command to be executed in the CAMAC Crate/Command Register (CNAF).

2.) Load the Transfer Count Register with the two's complement of the number of transfers to execute.

3.) Load the Control/Status Register (CSR) with M4, M2 and M1 set appropriately and GO set to a 1.

4.) Check the Outbound FIFO Flag in the Bus Master CSR to see if there is room in the FIFO for the CAMAC write data.

5.) If room exists for the write data, write the CAMAC write data into the FIFO Data Register.

6.) Check the DONE bit in the CSR to see if the interface has completed the CAMAC accesses or an error has occurred.

7.) If DONE is not set, jump back to step 4.

8.) If DONE is set check the ERR, No-X and No-Q bits in the CSR to verify data validity.

DMA BLOCK TRANSFERS

A Block Transfer Mode of operation transfers multiple CAMAC data words to/from a CAMAC crate. These modes provide a more efficient mechanism for transferring a block of data to/from a CAMAC module. This section describes the steps necessary to execute a block transfer operation using direct memory access (DMA) to move the data to/from the interface. This has an inherent speed advantage over programmed I/O transfers since the host processor is not involved with the actual moving of CAMAC write/read data from the interface.

When setting up DMA operations, the following PCI Interface Registers are accessed:

1.)	Master Write Address Register	Offset 24
2.)	Master Write Transfer Count Register	Offset 28
3.)	Master Read Address Register	Offset 2C
4.)	Master Read Transfer Count Register	Offset 30
5.)	Bus Master Control/Status Register	Offset 3C

The Master Write Address Register and Master Write Transfer Count Register are used to specify the DMA initial address and transfer byte count for CAMAC Read operations The WRITE references in the name of these registers refers to the actual direction of DMA, not the CAMAC access. The Master Read Address Register and Master Read Transfer Count Register are used to specify the DMA initial address and transfer byte count for CAMAC Write operations.

The Bus Master Control/Status Register is used to enable/disable the DMA transfers. The DMA transfers should not be enabled until after the GO bit has been set in the Control/Status Register. After the DMA operation and the CAMAC transfers are complete, the DMA transfers should then be disabled to prevent inadvertent data transfers to/from the FIFO Data Register for other operations.

To enable DMA transfers for CAMAC write operations, the READ TRANSFER ENABLE bit in the Bus Master Control/Status Register must be set to a one. For CAMAC read operations, the WRITE TRANSFER ENABLE bit in the Bus Master Control/Status Register must be set to a one.

The following sections illustrate the basic sequences necessary to setup and execute CAMAC block transfer write and read operations using DMA.

DMA Block Transfer Read Operation (F16 = 0 ; F8 = 0)

1.) Load the CAMAC Crate Address and the CAMAC read command to be executed in the CAMAC Crate/Command Register (CNAF).

2.) Load the Transfer Count Register with the two's complement of the number of transfers to execute.

3.) Load the Master Write Address Register with the initial memory address where CAMAC read data is to be stored.

4.) Load the Master Write Transfer Count Register with the maximum number of bytes that are to be transferred during the block transfer. Note that this is a byte count. Since the 2915 always executed 32-bit DMA operations, this value must be a multiple of 4.

5.) Set the WRITE TRANSFER ENABLE bit in the Bus Master Control/Status Register to a 1.

6.) Load the Control/Status Register (CSR) with M4, M2 and M1 set appropriately and GO set to a 1.

7.) Wait for the DONE bit in the CSR to be asserted.

8.) Check the ERR, No-X and No-Q bits in the CSR to verify data validity.

DMA Block Transfer Write Operation (F16 = 1 ; F8 = 0)

1.) Load the CAMAC Crate Address and the CAMAC read command to be executed in the CAMAC Crate/Command Register (CNAF).

2.) Load the Transfer Count Register with the two's complement of the number of transfers to execute.

3.) Load the Master Read Address Register with the initial memory address where CAMAC write data is stored.

4.) Load the Master Read Transfer Count Register with the maximum number of bytes that are to be transferred during the block transfer. Note that this is a byte count. Since the 2915 always executed 32-bit DMA operations, this value must be a multiple of 4.

5.) Load the Control/Status Register (CSR) with M4, M2 and M1 set appropriately and GO set to a 1.

6.) Set the READ TRANSFER ENABLE bit in the Bus Master Control/Status Register to a 1.

7.) Wait for the DONE bit in the CSR to be asserted.

8.) Check the ERR, No-X and No-Q bits in the CSR to verify data validity.

The following two functions written in 'C' are samples that illustrate setting up and executing DMA block transfers.

The first section shows a routine written to execute CAMAC block transfer read operations.

```
/*****************************************************************/
/*                                                               */
/* Filename : dmablkrd.c                                         */
/*                                                               */
/*****************************************************************/
/* Parameters:

                base1           -       base address register #1
                base2           -       base address register #2
                crate           -       CAMAC chassis to access
                n               -       CAMAC Station Number
                a               -       CAMAC Subaddress
                f               -       CAMAC Function Code
                q               -       returned CAMAC Q-response
                x               -       returned CAMAC X-response
                data            -       CAMAC read data array
                count           -       Transfer count
                qmode           -       Block Transfer Mode (1-4)
                word_size       -       CAMAC Data Word Size
                abort_disable   -       abort on NO-X bit                       */

#define ushort unsigned short
#define ulong unsigned long
unsigned long inpl(unsigned short address);

ushort dma_block_read(ushort base1,
                      ushort base2,
                      ushort crate,
                      ushort n,
                      ushort a,
                      ushort f,
                      ushort *q,
                      ushort *x,
                      ulong huge *data,
                      ulong count,
                      ushort qmode,
                      ushort word_size,
                      ushort abort_disable)
{
ushort csr = base2;                     /* define interface register addresses */
ushort naf = base2 + 0x04;
ushort tcr = base2 + 0x08;

ushort fifo = base1 + 0x20;             /* define PCI interface registers      */
ushort bmcsr = base1 + 0x3c;
ushort mwar = base1 + 0x24;
ushort mwtcr = base1 + 0x28;
```

```
ulong  mardata,byte_count,llp,ldata,stmo,csrdata,done;
ushort sdatalo;
short err;

ldata = ((ulong)(crate) << 16L) +          /* form CNAF register data                */
      ((ulong)(n) << 9L) +
      ((ulong)(a) << 5L) +
      ((ulong)(f));
outl(naf,ldata);                          /* write CNAF register                    */
ldata = ~(count-1);                       /* form 2's complement of transfer count  */
outl(tcr,ldata);                          /* write transfer count                   */

ldata = (ulong *)data;                    /* get physical memory address            */
mardata = ((ldata & 0xffff0000) >> 12) + (ldata & 0xffff);
outl(mwar,mardata);                       /* load write address register            */
if (word_size) {                          /* get actual byte counts                 */
 byte_count = count*2;
 }
else {
 byte_count = count*4;
 }
outl(mwtcr,byte_count);                   /* write transfer count register          */
ldata = 0x6000000;                        /* data for FIFO reset                    */
outl(bmcsr,ldata);                        /* reset fifo's                           */
ldata = 0x400;
outl(bmcsr,ldata);                        /* enable pci memory reads                */

ldata = ((ulong)(word_size) << 13L) +     /* form CSR data                          */
      ((ulong)(abort_disable) << 12L) +
      ((ulong)(qmode) << 1) + 1;
outl(csr,ldata);                          /* write csr                              */
stmo=0;                                   /* initialize timeout counter             */
while((stmo != 0x7ffff) && ((inpl(csr) & 0x80) != 0x80)) {     /* wait for done or timeout */
 stmo++;
 }
if (stmo == 0x7ffff) {
 printf("\n\nTimed out waiting for DONE .. dma_block_read\n");

 }
ldata = 0x6000000;                        /* write data                */
outl(bmcsr,ldata);                        /* reset fifo's & stop dma */
csrdata = inpl(csr);                      /* get status and return */
*q=1;
*x=1;
if ((csrdata & 0x10000) != 0) {           /* return CAMAC Q and X          */
 *q = 0;
 }
if ((csrdata & 0x20000) != 0) {
 *x = 0;
 }
sdatalo = (ushort)((csrdata & 0x80000000) >> 31);
return (sdatalo);
}
```

This section shows a routine written to execute CAMAC block transfer write operations.

```
/*****************************************************************/
/*                                                               */
/* Filename : dmablkwt.c                                         */
/*                                                               */
/*****************************************************************/
/* Parameters:

                base1          -       base address register #1
                base2          -       base address register #2
                crate          -       CAMAC chassis to access
                n              -       CAMAC Station Number
                a              -       CAMAC Subaddress
                f              -       CAMAC Function Code
                q              -       returned CAMAC Q-response
                x              -       returned CAMAC X-response
                data           -       CAMAC write data array
                count          -       Transfer count
                qmode          -       Block Transfer Mode (1-4)
                word_size      -       CAMAC Data Word Size
                abort_disable  -       abort on NO-X bit                */

#define ushort unsigned short
#define ulong unsigned long

unsigned long inpl(unsigned short address);

ushort dma_block_write(ushort base1,
                       ushort base2,
                       ushort crate,
                       ushort n,
                       ushort a,
                       ushort f,
                       ushort *q,
                       ushort *x,
                       ulong huge *data,
                       ulong count,
                       ushort qmode,
                       ushort word_size,
                       ushort abort_disable)
{
ushort csr = base2;                          /* define interface register addresses */
ushort naf = base2 + 0x04;
ushort tcr = base2 + 0x08;

ushort fifo = base1 + 0x20;                  /* define pci interface register addresses */
ushort bmcsr = base1 + 0x3c;
ushort mrar = base1 + 0x2c;
ushort mrtcr = base1 + 0x30;

ulong  mardata,byte_count,llp,ldata,stmo,csrdata,done;
ushort sdatalo;
short err;
```

```
ldata = ((ulong)(crate) << 16L) +                    /* form CNAF data          */
      ((ulong)(n) << 9L) +
      ((ulong)(a) << 5L) +
      ((ulong)(f));
outl(naf,ldata);                                    /* write cnaf */
ldata = ~(count-1);                                 /* form 2's complement of transfer count */
outl(tcr,ldata);                                    /* write transfer count    */

ldata = (ulong *)data;                              /* get physical buffer address        */
mardata = ((ldata & 0xffff0000) >> 12) + (ldata & 0xffff);
outl(mrar,mardata);                                 /* load read memory address           */
if (word_size) {                                    /* get actual byte count   */
 byte_count = count*2;
 }
else {
 byte_count = count*4;
 }
outl(mrtcr,byte_count);                             /* write read byte transfer count     */
ldata = 0x6000000;                                  /* write data                         */
outl(bmcsr,ldata);                                  /* reset fifo's */

ldata = ((ulong)(word_size) << 13L) +               /* form csr data                      */
      ((ulong)(abort_disable) << 12L) +
      ((ulong)(qmode) << 1L) + 1;
outl(csr,ldata);                                    /* write csr                          */
ldata = 0x4000;                                     /* write data                         */
outl(bmcsr,ldata);                                  /* enable pci memory reads            */
exit=0;
stmo=0;                                             /* wait for done or timeout           */
while((stmo != 0x7ffff) && ((inpl(csr) & 0x80) != 0x80)) {
 stmo++;
 }
if (stmo == 0x7ffff) {
 printf("\n\nTimed out waiting for DONE .. dma_block_write\n");

 }
ldata = 0x6000000;                                  /* write data                         */
outl(bmcsr,ldata);                                  /* reset fifo's & stop dma            */
csrdata = inpl(csr);                                /* read csr                           */
*q=1;                                               /* return CAMAC Q and X               */
*x=1;
if ((csrdata & 0x10000) != 0) {
 *q = 0;
 }
if ((csrdata & 0x20000) != 0) {
 *x = 0;
 }
sdatalo = (ushort)((csrdata & 0x80000000) >> 31);
return (sdatalo);
}
```

APPENDIX A

COMPOSITE REGISTER LAYOUT

Control/Status Register — Offset: 00 hex

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ERR	0	0	RST INFC	0	0	0	0	0	0	0	0	PBUS TMO	NAF TMO	NOX	NOQ
15	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**
0	0	WORD SIZE	ABT DIS	PCI IRQ	PCI IENA	RFS	RFS IENA	DONE	DONE IENA	CLR PCII	CLR DNI	MD 4	MD 2	MD 1	GO

CAMAC Command Register — Offset: 04 hex

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	C4	C2	C1
15	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**
0	0	N 16	N 8	N 4	N 2	N 1	A 8	A 4	A 2	A 1	F 16	F 8	F 4	F 2	F 1

Transfer Count Register — Offset: 08 hex

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	TC 23	TC 22	TC 21	TC 20	TC 19	TC 18	TC 17	TC 16
15	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**
TC 15	TC 14	TC 13	TC 12	TC 11	TC 10	TC 09	TC 08	TC 07	TC 06	TC 05	TC 04	TC 03	TC 02	TC 01	TC 00

Service Request Register — Offset: 0C hex

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	**14**	**13**	**12**	**11**	**10**	**09**	**08**	**07**	**06**	**05**	**04**	**03**	**02**	**01**	**00**
0	0	0	0	0	0	0	0	C7	C6	C5	C4	C3	C2	C1	C0

WARRANTY

KineticSystems Company, LLC warrants its standard hardware products to be free of defects in workmanship and materials for a period of one year from the date of shipment to the original end user. Software products manufactured by KineticSystems are warranted to conform to the Software Product Description (SPD) applicable at the time of purchase for a period of ninety days from the date of shipment to the original end user. Products purchased for resale by KineticSystems carry the original equipment manufacturer's warranty.

KineticSystems will, at its option, either repair or replace products that prove to be defective in materials or workmanship during the warranty period.

Transportation charges for shipping products to KineticSystems shall be prepaid by the purchaser, while charges for returning the repaired warranty product to the purchaser, if located in the United States, shall be paid by KineticSystems. Return shipment will be made by UPS, where available, unless the purchaser requests a premium method of shipment at their expense. The selected carrier shall not be construed to be the agent of KineticSystems, nor will KineticSystems assume any liability in connection with the services provided by the carrier.

The product warranty may vary outside the United States and does not include shipping, customs clearance, or any other charges. Consult your local authorized representative or reseller for more information regarding specific warranty coverage and shipping details.

PRODUCT SPECIFICATIONS AND DESCRIPTIONS IN THIS DOCUMENT SUBJECT TO CHANGE WITHOUT NOTICE.

KINETICSYSTEMS SPECIFICALLY MAKES NO WARRANTY OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE OR ANY OTHER WARRANTY EITHER EXPRESSED OR IMPLIED, EXCEPT AS IS EXPRESSLY SET FORTH HEREIN. PRODUCT FAILURES CREATED BY UNAUTHORIZED MODIFICATIONS, PRODUCT MISUSE, OR IMPROPER INSTALLATION ARE NOT COVERED BY THIS WARRANTY.

THE WARRANTIES PROVIIDED HEREIN ARE THE PURCHASER'S SOLE AND EXCLUSIVE REMEDIES ON ANY CLAIM OF ANY KIND FOR ANY LOSS OR DAMAGE ARISING OUT OF, CONNECTED WITH, OR RESULTING FROM THE USE, PERFORMANCE OR BREACH THEREOF, OR FROM THE DESIGN, MANUFACTURE, SALE, DELIVERY, RESALE, OR REPAIR OR USE OF ANY PRODUCTS COVERED OR FURNISHED BY KINETICSYSTEMS INCLUDING BUT NOT LIMITED TO ANY CLAIM OF NEGLIGENCE OR OTHER TORTIOUS BREACH, SHALL BE THE REPAIR OR REPLACEMENT, FOB FACTORY, AS KINETICSYSTEMS MAY ELECT, OF THE PRODUCT OR PART THEREOF GIVING RISE TO SUCH CLAIM, EXCEPT THAT KINETICSYSTEMS' LIABILITY FOR SUCH REPAIR OR REPLACEMENT SHALL IN NO EVENT EXCEED THE CONTRACT PRICE ALLOCABLE TO THE PRODUCTS OR PART THEROF WHICH GIVES RISE TO THE CLAIM. IN NO EVENT SHALL KINETICSYSTEMS BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES, INCLUDING LOSS OF PROFITS.

Products will not be accepted for credit or exchange without the prior written approval of KineticSystems. If it is necessary to return a product for repair, replacement or exchange, a Return Authorization (RA) Number must first be obtained from the Repair Service Center prior to shipping the product to KineticSystems. The following steps should be taken before returning any product:

1. Contact KineticSystems and discuss the problem with a Technical Service Engineer.
2. Obtain a Return Authorization (RA) Number.
3. Initiate a purchase order for the estimated repair charge if the product is out of warranty.
4. Include a description of the problem and your technical contact person with the product.
5. Ship the product prepaid with the RA Number marked on the outside of the package to:

KineticSystems Company, LLC
Repair Service Center
900 North State Street
Lockport, IL 60441

Telephone: (815) 838-0005
Facsimile: (815) 838-4424
Email: tech-serv@kscorp.com