

ARTISAN®
TECHNOLOGY GROUP

Your **definitive** source for quality pre-owned equipment.

Artisan Technology Group
(217) 352-9330 | sales@artisantg.com | artisantg.com

Full-service, independent repair center
with experienced engineers and technicians on staff.

We buy your excess, underutilized, and idle equipment
along with credit for buybacks and trade-ins.

Custom engineering
so your equipment works exactly as you specify.

- Critical and expedited services
- In stock / Ready-to-ship
- Leasing / Rentals / Demos
- ITAR-certified secure asset solutions

Expert team | Trust guarantee | 100% satisfaction

All trademarks, brand names, and brands appearing herein are the property of their respective owners.

Visit our website - *Click HERE*

scribed in the Operation Control Register description.

Only the least significant 24 bits of the counter (MACMH, MACML, and MACL) are implemented in the DMAI. The most significant byte of the counter, MACH, is provided only to allow compatibility with programming of the 68440 and 68450. Writing to MACH has no effect on the DMAI operation. Reading MACH always returns H'00'.

Memory Transfer Counter (MTCH, MTCL)

The 16-bit memory transfer counter programs the number of operands to be transferred by the DMAI. The counter must be initialized prior to beginning the transfer of a block of data and then decrements once per operand transfer (regardless of operand size) until it reaches the terminal value of zero. Channel operation then terminates and the COC bit in the CSR will be asserted.

Interrupt Vector Register (IVR)

The IVR contains the value to be placed on the data bus upon receipt of an interrupt acknowledge from the MPU. Only the seven most significant bits of the programmed value are used by the DMAI. The output vector from the DMAI contains a zero in the least significant bit position if a normal termination occurred (error bit not set) and contains a one in the least significant bit position if termination was due to an error (error bit set).

The contents of this register are initialized to H'0F' by a reset. The value returned will be H'0F', regardless of the error state, until the register is programmed by the MPU.

To provide compatibility with the other 68K family DMA controllers, the IVR has two addresses (see table 1). If program compatibility is required, the value written at the normal IVR address should have a zero as its LSB, and the value written at the error IVR address should be the same but with the LSB equal to one.

OPERATION

A DMAI operation proceeds in three principal phases. During the initialization phase, the MPU configures the channel control registers, loads the initial memory address and transfer count, and starts the channel. During the transfer phase, the DMAI accepts requests for transfers from the device, arbitrates for and acquires ownership of the bus, and provides for addressing and bus controls for the transfers. The termination phase occurs after the operation is complete, when the DMAI reports the status of the operation.

Operation Initiation

After having programmed the control registers, the memory address counter, and the memory transfer counter, the MPU sets the start bit (CCR[7]). The DMAI initiates the operation by clearing any pending requests, clearing the start bit, and setting the channel active bit in the CSR. The DMAI is then ready to receive valid requests for an operation.

The channel cannot be started if any of the internal status bits in the CSR (CSR[15:11]) have not been cleared. An error is not signaled if this condition occurs. The only indication of this state is that the start bit remains set in the CCR. A pending start cannot be reset by a write to the register. START can be cleared only by the DMAI when it starts operation or by setting the software abort bit (CCR[4]).

Device/DMAI Communication

Communication between the peripheral device and the DMAI is accommodated by five signal lines:

Request (REQN)

The device makes a request for service by asserting the request line. The DMAI can operate in either the burst request mode or the cycle stealing request mode, as programmed by the external request mode bit (DCR[15]).

The burst mode allows a device to request the transfer of multiple operands using consecutive bus cycles. In this mode the request line is an active low input. The DMAI services the request by arbitrating for the bus, obtaining the bus, and notifying the peripheral by asserting the acknowledge (ACKN) output. If the request line is active when the DMAI asserts ACKN, and remains active at least until the DMAI asserts device transfer complete (DTCN), the DMAI recognizes a valid request for another operand, which will be transferred during the next bus cycle. If the request line is negated before the DMAI asserts DTCN, the DMAI relinquishes the bus and waits for the next request. For long word transfers (2 x 16), the request must be asserted at least until the acknowledge for the second part of the operand has been asserted.

In the cycle steal mode, the device requests an operand transfer by generating a falling edge on the request line. The DMAI services the request by arbitrating for the bus, obtaining the bus, and notifying the peripheral by asserting the acknowledge (ACKN) output. The request line must be in the inactive state for at least one clock cycle before a request is made. After a request has been asserted, it must remain at the assertion level for at least one clock cycle. If another request is received before the first operand part of a former request is acknowledged, the second request is not recognized. Normally, the DMAI will relinquish the bus after servicing a valid request. However, if the device generates a new request before the DMAI asserts DTCN for the last operand part, the DMAI will retain ownership of the bus and that request will be serviced before the bus is relinquished.

Acknowledge (ACKN)

The DMAI asserts the acknowledge line, which implicitly addresses the device making the request, during transfers to and from the device. The line may be used to control buffering circuits between the data bus and the MPU bus.

During burst mode, REQN must not be disasserted for less than one CLK period plus four RC time constants, where R is the value of the resistor used for the pullup on BRN and C has a typical value of 20pF.

Ready (RDYN)

Ready is an active low input which is asserted by the requesting device to indicate to the DMAI that valid data has either been stored or put on the bus. If negated, it indicates that the data has not been stored or presented, causing the DMAI to enter wait states until RDYN is asserted. RDYN can be held low continuously if the device is fast enough so that wait states are not required. The current state of the ready input is reflected in CSR[8].

Done (DONEN)

Done is a bidirectional active low signal. As an output, it is asserted and negated by the DMAI concurrent with the ACKN output of the last operand part to indicate to the device that the memory transfer count is exhausted and that the DMAI's operation is completed as a result of that transfer.

The DMAI also monitors the state of the line while acknowledging a device. If the device asserts DONEN, the DMAI will terminate operation after the transfer of the current operand. In this case the DMAI clears the channel active bit and sets the channel operation complete and normal device termination bits in the CSR. If both the DMAI and the device assert DONEN, the device termination is not recognized, but the operation does terminate.

Device Transfer Complete (DTCN)

DTCN is an active low output which is asserted by the DMAI to indicate to the device that the requested data transfer is complete. On a write to memory operation, it indicates that the data provided by the device has been successfully stored. On a read from memory operation, it indicates to the device that the data from memory is present on the data bus and should be latched. DTCN is not asserted if assertion of the RERUNN input terminates the bus cycle.

Bus Arbitration

Upon receiving a valid request for a transfer from the device, the DMAI will arbitrate for and obtain ownership of the system bus.

The DMAI indicates that it wishes to become the bus master by asserting its bus request (BRN) output. This is a wire-ORed signal that indicates to the MPU that some external device requires control of the bus. The processor is effectively at a lower priority level than external devices and will relinquish the bus after it has completed the last bus cycle it has started. The processor puts the bus up for external arbitration by asserting its bus grant (BGN) output. This signal may be routed through a daisy chain (such as provided by the DMAI) or through some other priority-encoded network. When the DMAI making the bus request receives the bus grant (indicated by its BGN input being asserted), it is to be the next bus master. It waits until address strobe (ASN), data transfer acknowledge (DTACKN) and bus grant acknowledge (BGACKN) become inactive and then assumes ownership of the bus by asserting its own BGACKN output. The DMAI then negates the BRN output and proceeds with the data transfer phase. After this phase is completed, the DMAI relinquishes bus ownership by negating the BGACKN output.

In burst DMA mode, detection of an active low request input after the DMAI operation has been started will begin the bus arbitration cycle. However, if the device negates its request at least one clock cycle before the DMAI asserts BGACKN, the DMAI will negate its bus request and will not assume ownership of the bus.

Data Transfers

The actual transfer of data between the memory and the device occurs during the data transfer phase. All transfers occur during a single cycle except in the case of long word operands, in which case two cycles are used to transfer the operand as two 16-bit words. The transfers take place using a 'single address' protocol; the DMAI addresses the memory via the bus address lines, while the device is implicitly addressed via the acknowledge output.

When a request is generated using the request method programmed in the control register, the DMAI obtains the bus and asserts acknowledge to notify the device that a transfer is to take place. The DMAI asserts all S68000 bus control signals needed for the transfer and holds them until the device responds with ready. The bus cycle then terminates normally. Ready may be tied low (asserted) if the device is fast enough.

When the transfer is from memory to the device, data is valid when DTACKN is asserted by the memory and remains valid until the data strobe(s) are negated. The assertion of DTCN from the DMAI can be used to latch the data, as the data strobes are not removed until one-half clock after the assertion of DTCN.

When the transfer is from device to memory, the data must be valid on the bus before the DMAI asserts the data strobe(s). The device indicates valid data by asserting ready. The DMAI then asserts the strobes and holds them asserted until the memory accepts the data, indicated by the assertion of DTACKN. The DMAI then asserts DTCN and negates the data strobes.

Flow charts for these operations are shown in figures 1 and 2. Refer to the timing section for the equivalent timing diagrams.

Operation Termination

Termination of the block transfer occurs under the conditions detailed below.

Terminal Count

As part of each transfer of an operand, the DMAI decrements the memory transfer counter. If this counter is decremented to zero, the operand is the last operand of the block. The DMAI operation is complete and it notifies the device of completion by asserting the DONEN output during the last operand transfer cycle. When the transfer has been completed, the channel active bit in the CSR is cleared and the COC bit is set, unless an error occurs.

Device Termination

The DMAI monitors the state of the DONEN line while acknowledging a device transfer request. If the device asserts DONEN, the DMAI will terminate operation after the transfer of the current operand. When the transfer has been completed, the DMAI clears the channel active bit and sets the COC and normal device termination bits in the CSR. If both the DMAI and the device assert DONEN, the device termination is not recognized, but the operation does terminate.

Software Abort

The software abort bit (CCR[4]) allows the MPU to abort the current operation of the DMAI. The COC and error bits in the CSR are set, the channel active bit in the CSR is cleared, and an abort error condition is signaled in the CER.

Rerun Error

The DMAI provides a rerun input (RERUNN) to indicate a bus exception condition. RERUNN must arrive prior to or in coincidence with DTACKN in order to be recognized, and the DMAI verifies that the line has been stable for two clock cycles before acting on it. The occurrence of a rerun during a DMAI bus cycle forces it to terminate the bus cycle in an orderly manner.

When the assertion of rerun is verified, the DMAI stops operation and three-states the data, address, and control lines, except BGACKN, so that it retains ownership of the bus. It remains halted until rerun becomes inactive, and then re-tries the last bus cycle. If rerun is asserted again, the DMAI stops DMA operation, releases the bus, sets the error and COC bits in the CSR, clears the active bit in the CSR, and sets the error code in the CER to indicate a bus error.

While stopped due to assertion of rerun, the DMAI does not generate any bus cycles and will not honor any requests until it is removed. However, the DMAI still recognizes requests.

Error Recovery Procedure

If an error occurs during a DMA transfer such that the DMAI stops the DMA operation, information is available to the operating system for an error recovery routine.

The information available to the operating system consists of the memory address counter, the memory transfer counter, and the control, status, and error registers. The DMAI decrements the memory transfer counter before attempting a DMA operation, so the register will contain the count minus one of the attempted transfer. The memory address counter will contain the address at which the DMA operation was attempted.

Reset

The reset input (RESETN) provides a means of resetting and initializing the DMAI from an external source. If the DMAI is a bus master when reset is received, the DMAI relinquishes the bus. Reset clears the control and error registers, sets all bits of the status register except CSR[8] to zero, and initializes the interrupt vector register to H'0'F.

Interrupts

The interrupt enable bit (CCR[3]) determines whether the DMAI generates interrupt requests. When the bit is set, an interrupt request is generated if the channel operation complete bit in the CSR is set. When the IACKN input is asserted, and the DMAI has an interrupt request pending, the DMAI returns an interrupt vector on the data bus.

The interrupt vector issued is the contents of the IVR. Only the seven most significant bits of the programmed value are used by the DMAI. The vector from the DMAI contains a zero in the LSB position if a normal termination occurred (error bit not set) and contains a one in the LSB position if termination was due to an error (error bit set).

The contents of this register are initialized to H'0F' by a reset. The value returned will be H'0F', regardless of the error state, until the register is programmed by the MPU.

To provide compatibility with the other 68K family DMA controllers, the IVR has two addresses (see table 1). If program compatibility is required, the value written at the

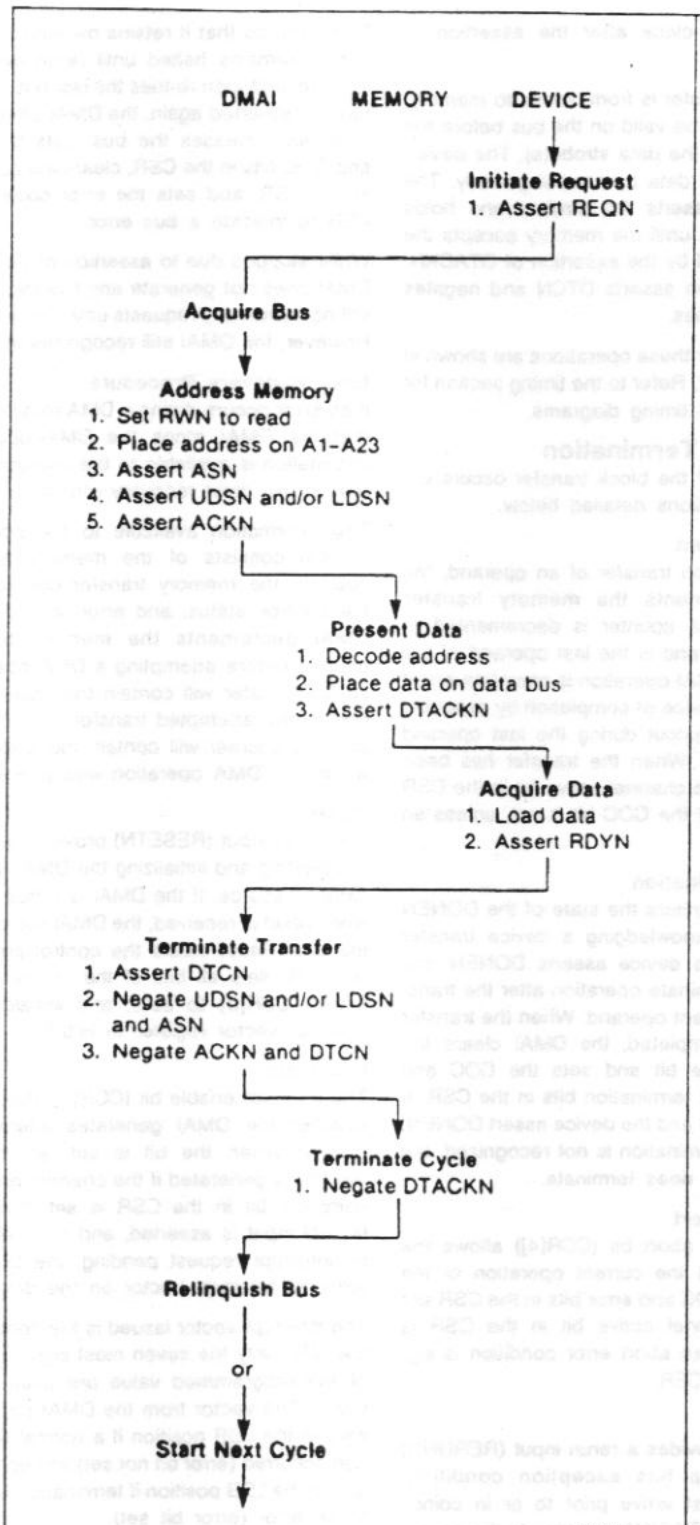


Figure 1. Transfer from Memory to Device Flowchart

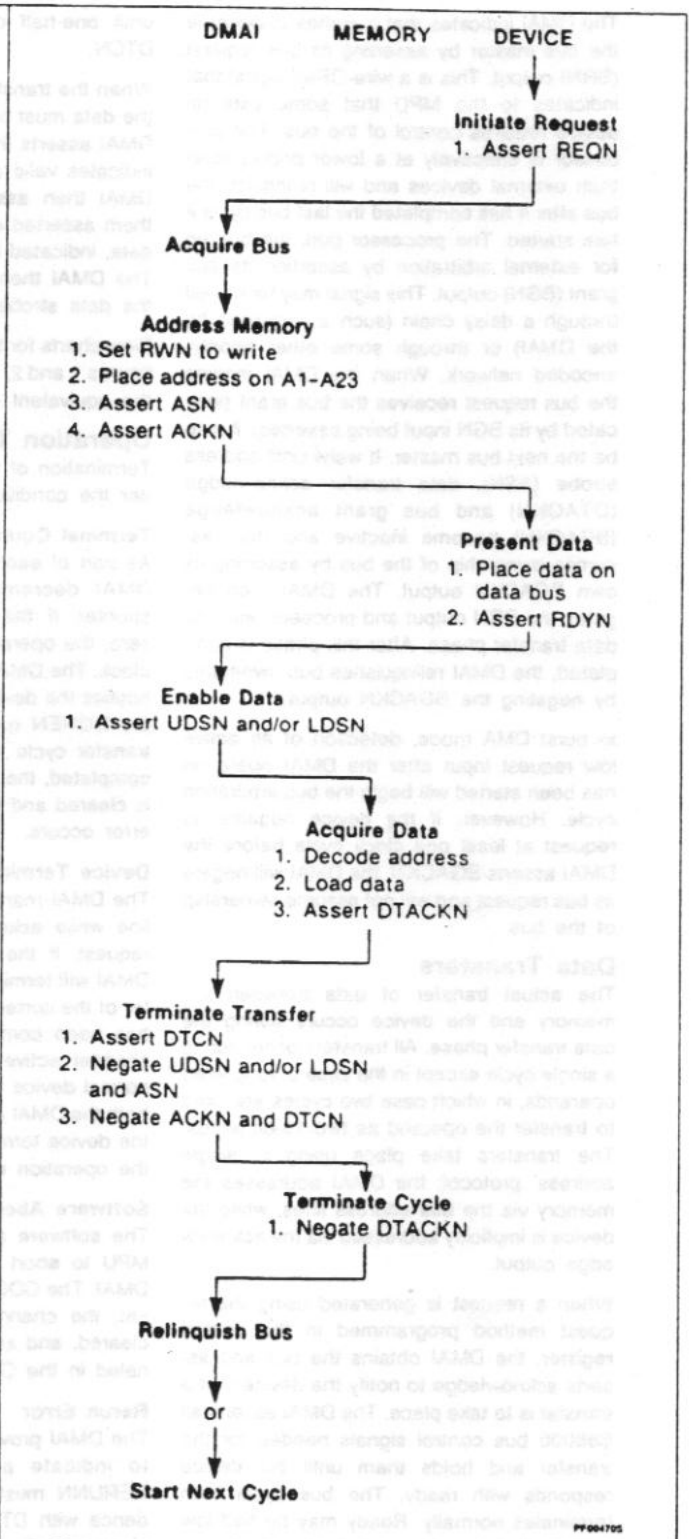


Figure 2. Transfer from Device to Memory Flowchart

normal IVR address should have a zero as its LSB, and the value written at the error IVR address should be the same but with the LSB equal to one.

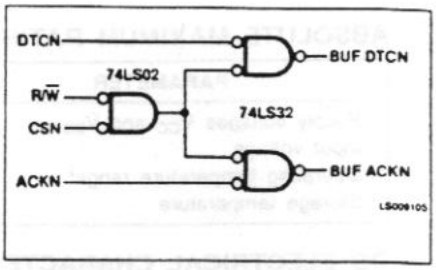
APPLICATIONS

Figure 3 illustrates a typical interconnection of the DMAI in a 68000 based system.

DESIGN NOTE

When clearing the error bit in CSR (bit 12) after a DMAI abort due to a double RERUNN, ACKN and DTCN will both go low concurrent with CSN and DTACKN for one CLK cycle.

To prevent the possibility that the device may misinterpret these signals, it is suggested that these signals be ANDed with CSN (see figure below).



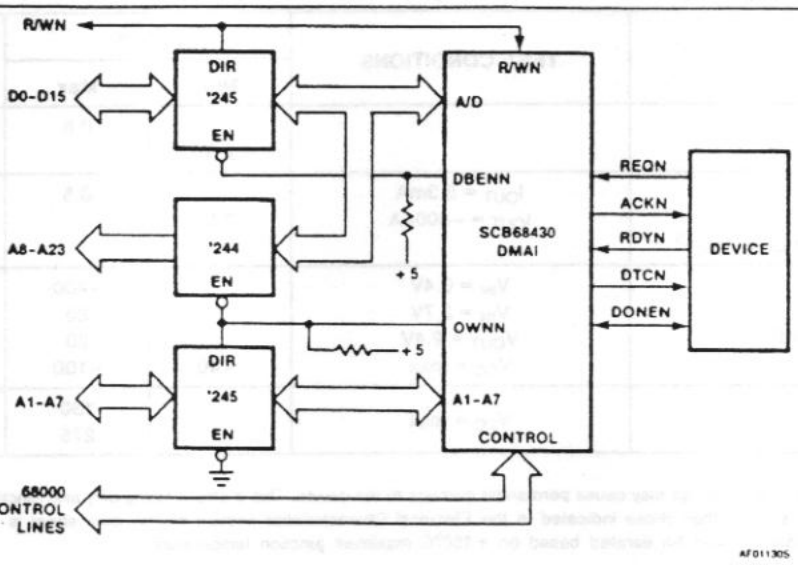


Figure 3. DMAI Application

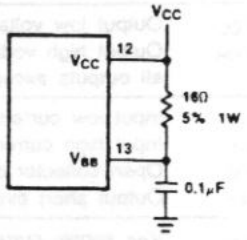


Figure 4. Recommended V_{BB} Test Circuit

ABSOLUTE MAXIMUM RATINGS[1]

PARAMETER	RATING	UNIT
Supply voltages V_{CC} and V_{BB}	−0.5 to +7.0	V
Input voltage	−0.5 to +5.5	V
Operating temperature range[2]	0 to +70	°C
Storage temperature	−65 to +150	°C

DC ELECTRICAL CHARACTERISTICS V_{CC} = 5.0V ±5%, V_{BB} = Figure 4, T_A = 0°C to +70°C[3,4,7]

PARAMETER		TEST CONDITIONS	LIMITS		UNIT
			Min	Max	
V_{IL}	Input low voltage			0.8	V
V_{IH}	Input high voltage		2.0		V
V_{OL}	Output low voltage	I_{OUT} = 5.3mA		0.5	V
V_{OH}	Output high voltage, all outputs except open collector outputs[5]	I_{OUT} = −400μA	2.5		V
I_{IL}	Input low current	V_{IN} = 0.4V		−400	μA
I_{IH}	Input high current	V_{IN} = 2.7V		20	μA
I_{OC}	Open collector off state current[5]	V_{OUT} = 2.4V		20	μA
I_{SC}	Output short circuit current[6]	V_{CC} = max	−40	−100	mA
I_{CC}	V_{CC} supply current	V_{CC} = max		130	mA
I_{BB}	V_{BB} supply current			275	mA

NOTES:

1. Stresses above those listed under Absolute Maximum Ratings may cause permanent damage to the device. This is stress rating only and functional operation of the device at these or at any other conditions other than those indicated in the Electrical Characteristics section of this data sheet is not implied.
2. For operating at elevated temperatures, the device must be derated based on +150°C maximum junction temperature.
3. Parameters are valid over specified temperature range.
4. All voltage measurements are referenced to ground (V_{SS}). For testing, all signals swing between 0.4V and 2.4V with a transition time of 20ns maximum. All time measurements are referenced at input voltages of 0.8V and 2.0V as appropriate.
5. IRQN, BRN, DONEN, and OWNN are open collector outputs.
6. No more than one output should be connected to ground at one time.
7. Capacitive test load is 100pF for all pins except DTCN which has a 35pF capacitive test load.

AC ELECTRICAL CHARACTERISTICS V_{CC} = 5.0V ±5%, V_{BB} = Figure 4, T_A = 0°C to +70°C[3,4,7]

NO.	FIGURE	CHARACTERISTICS	TENTATIVE LIMITS				UNIT
			10MHz		12.5MHz		
			Min	Max	Min	Max	
1	5	A1 – A7, ASN, RWN, set-up to UDSN, LDSN low	0		0		ns
2	5	D0 – D15 3-state to invalid data from ASN, CSN, and UDSN or LDSN low	10		10		ns
3	5	DTACKN 3-state to high from ASN, CSN, and UDSN or LDSN low	10		10		ns
4	5	CSN low after UDSN or LDSN low		25		25	ns
5	5, 6	DBENN low after ASN and CSN low		60		60	ns
6	5	D0 – D15 valid data from ASN, CSN, and UDSN or LDSN low		100		100	ns
7	5	DTACKN low after D0 – D15 valid data	−15	30	−15	30	ns
8	5	A1 – A7, ASN, RWN or CSN hold after UDSN and LDSN high	0		0		ns
9	5, 6	DBENN high from either ASN or CSN high		45		45	ns
10	5	D0 – D15 to 3-state from UDSN and LDSN high		80		80	ns
11	5	D0 – D15 to invalid data from UDSN and LDSN high	10		10		ns
12	5, 6	DTACKN high from UDSN and LDSN high		55		55	ns
13	5, 6	DTACK 3-state from either CSN or ASN high		85		85	ns
14	6	A1 – A7, ASN, RWN set-up to UDSN, LDSN low	50		50		ns/s
15	6	CSN set-up before UDSN or LDSN low	20		20		ns
16	6	DTACKN 3-state to high after CSN and ASN low	10		10		ns
17	6	D0 – D15 valid after UDSN or LDSN low		0		0	ns
18	6	DTACKN low from UDSN or LDSN low		100		100	ns
19	6	UDSN and LDSN low time	115		100		ns
20	6	A1 – A7 hold after UDSN and LDSN high	0		0		ns
21	6	ASN, RWN and CSN hold after UDSN and LDSN high	0		0		ns

AC ELECTRICAL CHARACTERISTICS (Continued)

NO.	FIGURE	CHARACTERISTICS	TENTATIVE LIMITS				UNIT
			10MHz		12.5MHz		
			Min	Max	Min	Max	
22	6	D0 – D15 hold after UDSN and LDSN high	0		0		ns
23	7	DBENN low from last low of ASN, IACKN, LDSN		65		65	ns
24	7	D0 – D7 valid after last low of ASN, IACKN, LDSN		105		105	ns
25	7	DTACKN 3-state to high after last low of ASN, IACKN, LDSN		100		100	ns
26	7	DTACKN low after last low of ASN, IACKN, LDSN		110		110	ns
27	7	DBENN high after first high of ASN, IACKN, LDSN		55		55	ns
28	7	D0 – D7 hold after first high of ASN, IACKN, LDSN		60		60	ns
29	7	D0 – D7 3-state after first high of ASN, IACKN, LDSN		80		80	ns
30	7	DTACKN high after first high of ASN, IACKN, LDSN		60		60	ns
31	7	DTACKN 3-state after first high of ASN, IACKN, LDSN		95		95	ns
32	8	BRN high from CLK high		65		65	ns
33	8, 11, 12	BGACKN low from CLK low		75		75	ns
34	8, 11, 12	OWNN low from CLK high		75		75	ns
35	8	BGACKN high from CLK low		75		75	ns
36	8,11,12	OWNN high from CLK high (load dependent)		50		50	ns
37	10	REQN set-up before CLK low	30		30		ns
38	10	REQN hold after CLK high	20		20		ns
39	10	BRN low from CLK high		80		80	ns
41	11, 12	ASN, UDSN, LDSN, RWN 3-state to high from CLK low		75		75	ns
43	11, 12	A1 – A23 to valid ASN	0		0		ns
44	11, 12	ASN low from CLK high		65		65	ns
45	11, 12	LDSN, UDSN low from CLK high		90		90	ns
46	11, 12	ACKN low from CLK high		65		65	ns
47	11, 12	DTACKN set-up to CLK high	30		30		ns
48	11, 12	RDYN set-up to CLK low	30		30		ns
49	11, 12	DTCN low from CLK high		70		70	ns
50	11, 12	ASN high from CLK high		75		75	ns
51	11, 12	LDSN, UDSN, high from CLK high		90		90	ns
52	11, 12	DTACKN, RDYN hold after CLK high	0		0		ns
–	11, 12	ASN, LDSN, UDSN, high from DTCN low	–20		–20		ns
53	11, 12	ACKN high from CLK high		50		50	ns
54	11, 12	DTCN high from CLK high		50		50	ns
55	11, 12	Address valid after CLK low	10		10		ns
–	11, 12	Address valid after ASN high	0		10		ns
56	11, 12	DONEN (output) low from CLK low		120		120	ns
57	11, 12	DONEN (output) high from CLK high		50		50	ns
58	11, 12	DONEN (input) set-up low before CLK low	30		30		ns
59	11, 12	DONEN (input) hold low after CLK high	0		0		ns
60	11, 12	BGACKN, ASN, UDSN, LDSN, RWN to 3-state from CLK low		75		75	ns
62	11, 12	A1 – A23 valid to 3-state from CLK high		100		100	ns
63	12	R/WN low from CLK high		65		65	ns
64	12	R/WN high from CLK high		75		75	ns
65	13	RERUNN set-up low before CLK high	30		30		ns
66	13	RERUNN hold low from CLK high	20		20		ns
67	13	A1 – A23 to idle state from CLK low		100		100	ns
68	13	A1 – A23 to valid after CLK low		85		85	ns

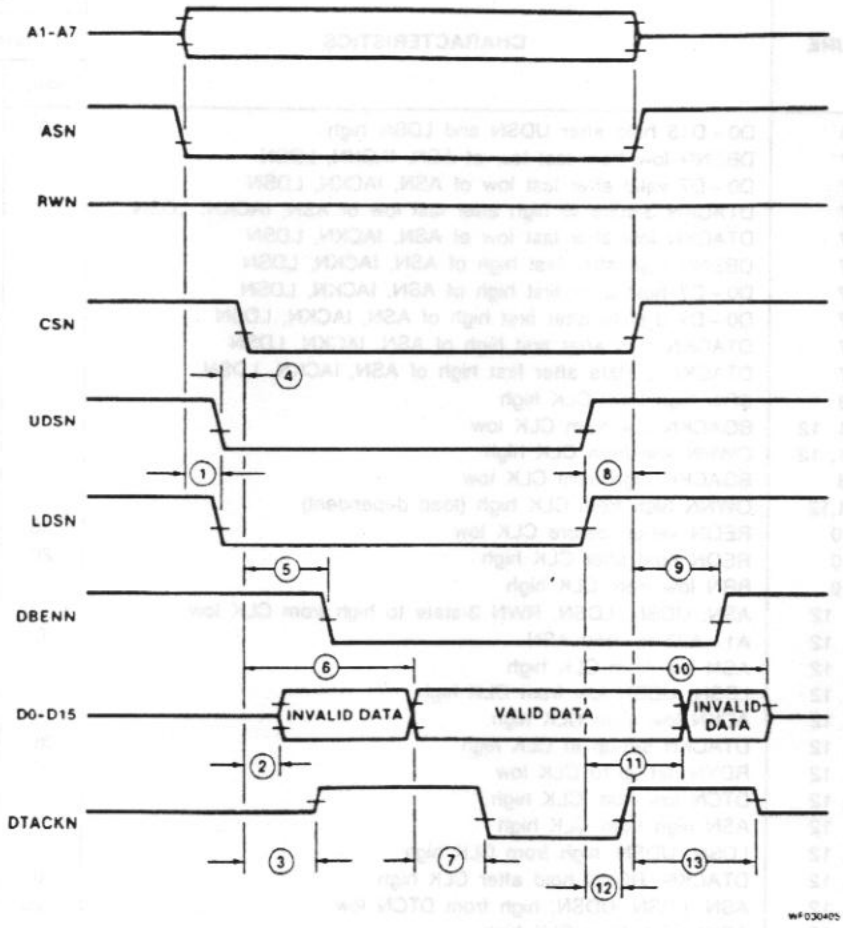


Figure 5. DMAI Read Timing

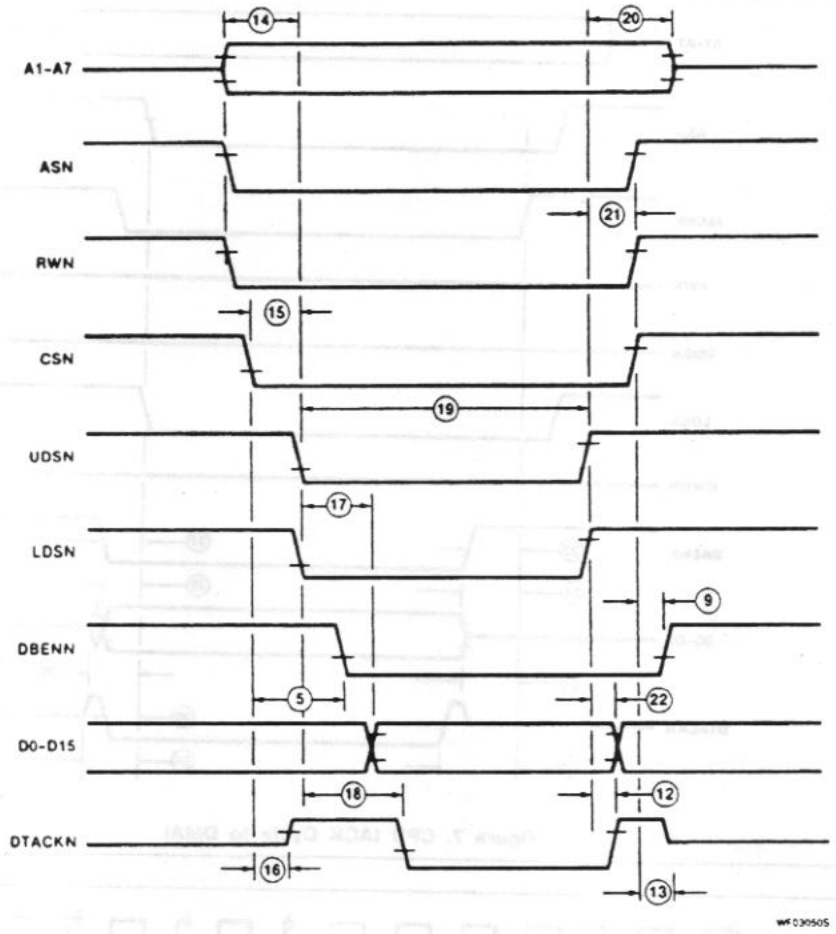


Figure 6. DMAI Write Timing

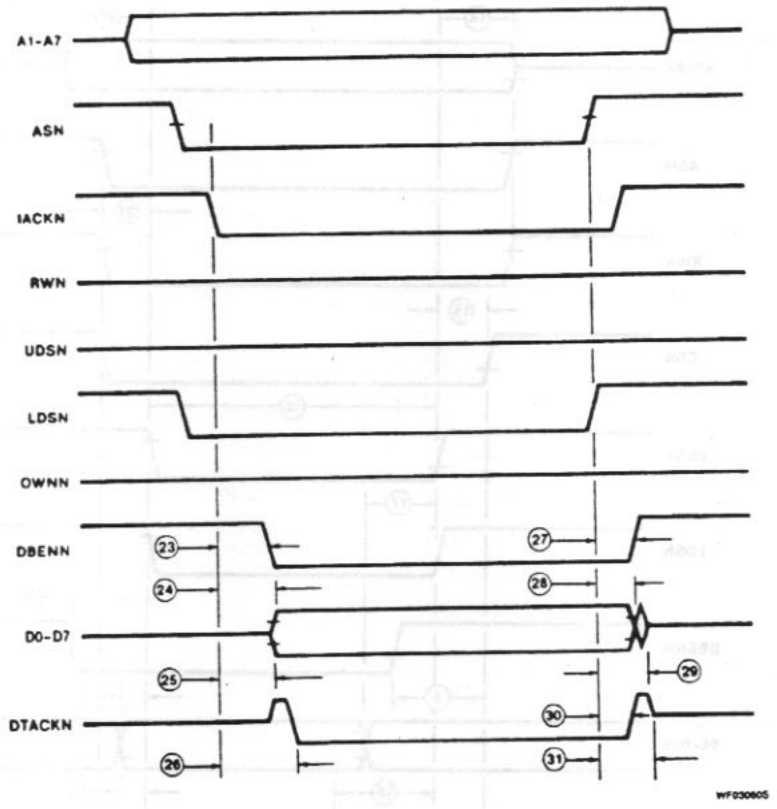
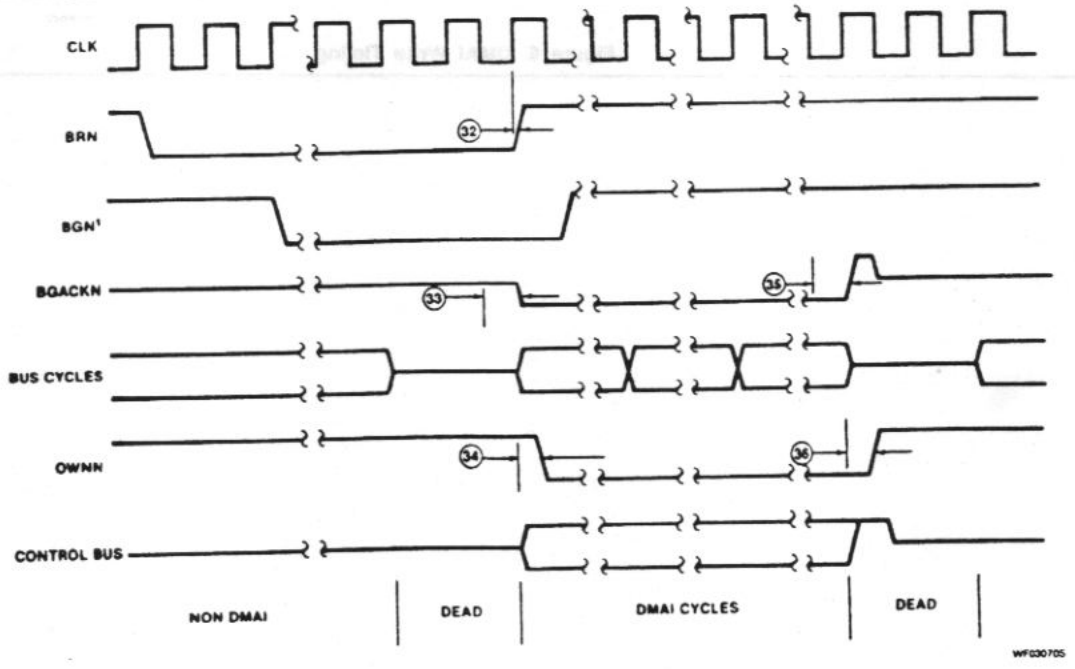


Figure 7. CPU IACK Cycle to DMAI



NOTES:
1. Device will become master if BGN is asserted concurrent with or later than REQN (same clock edge or later).
2. ASN, DTACKN and BGACKN must be negated in order for DMAI to become master. Timing assumes all these happen concurrent with BGN — if not, it is from the latest signal which is negated.

Figure 8. DMAI Bus Arbitration Timing

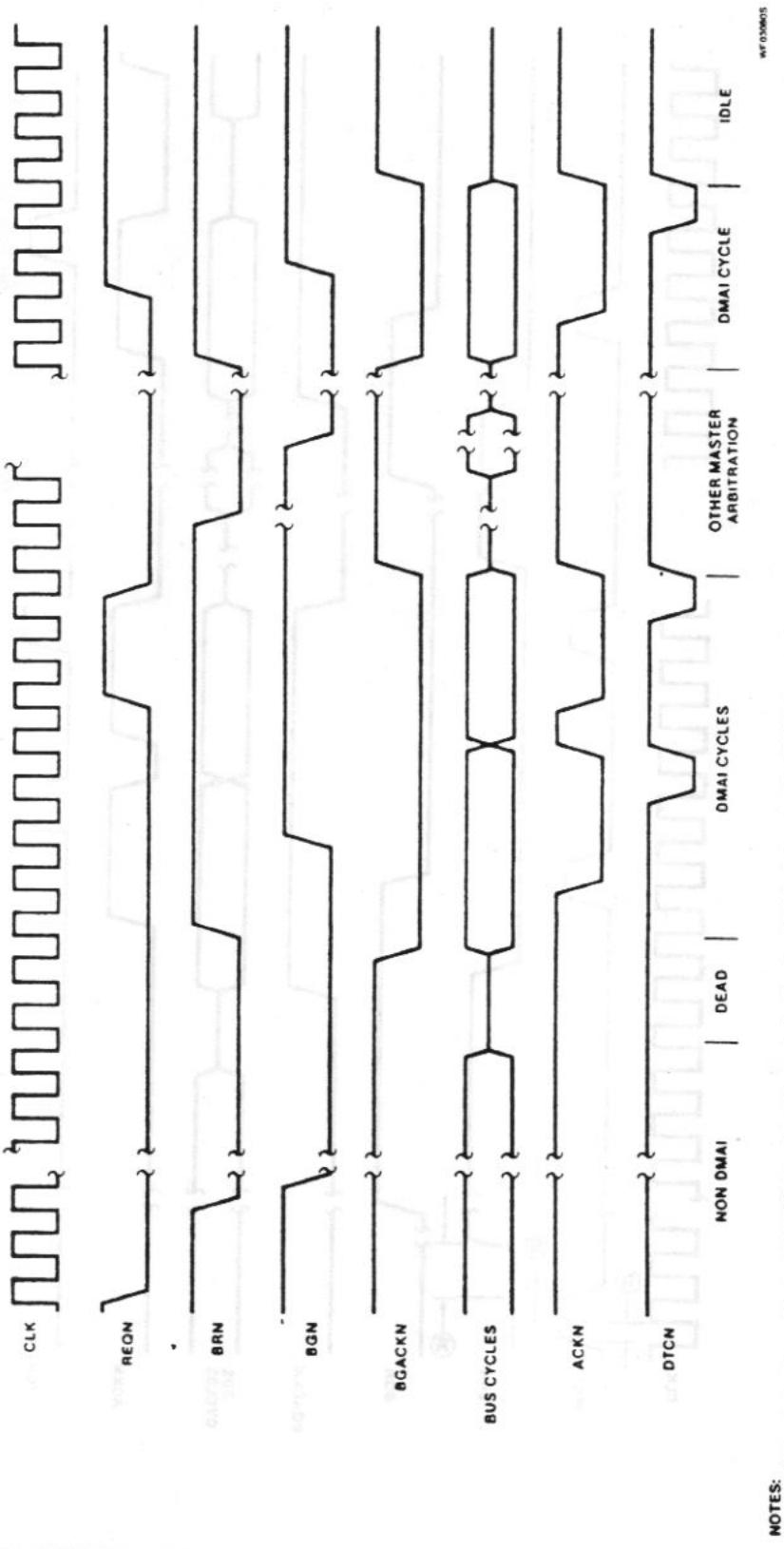
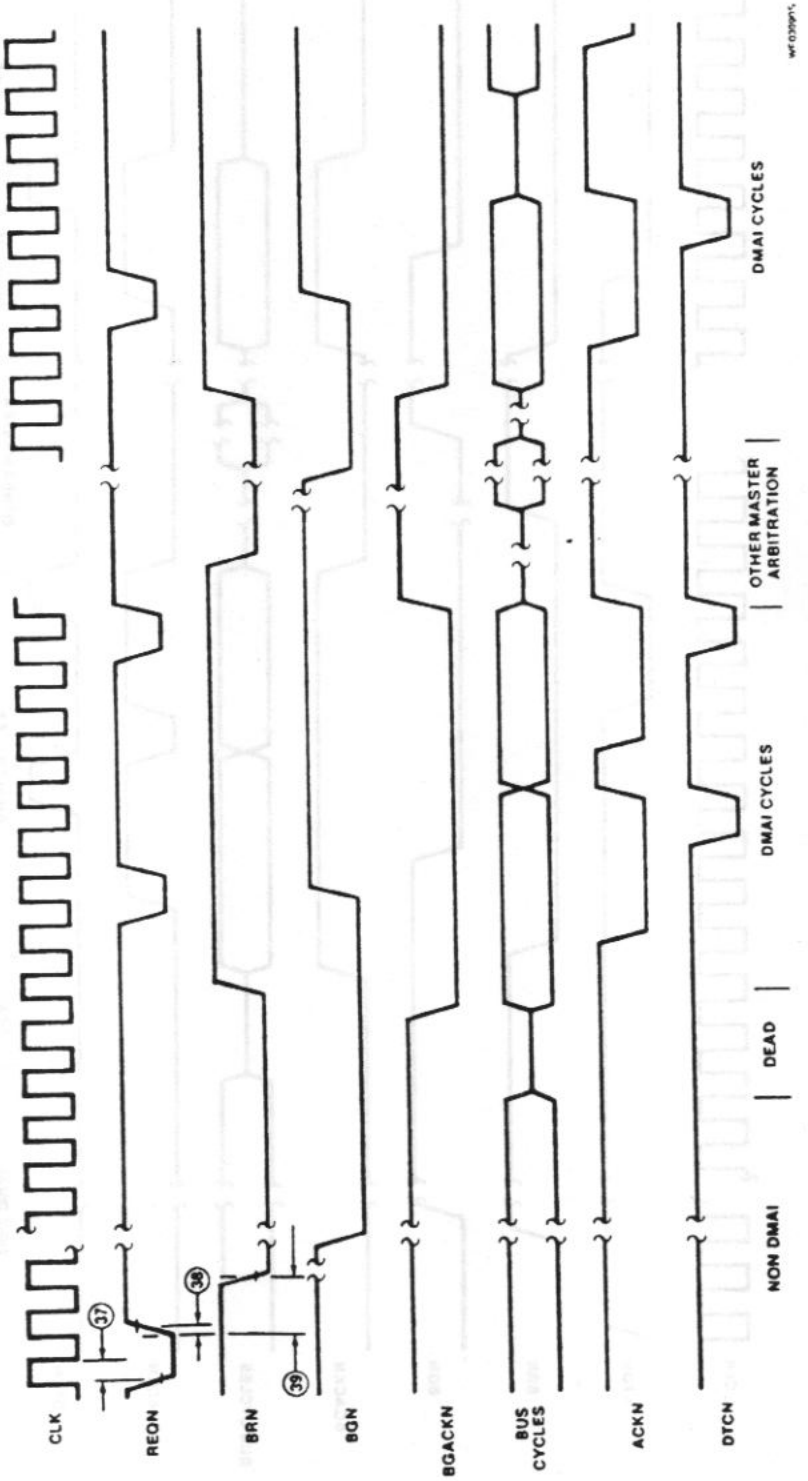


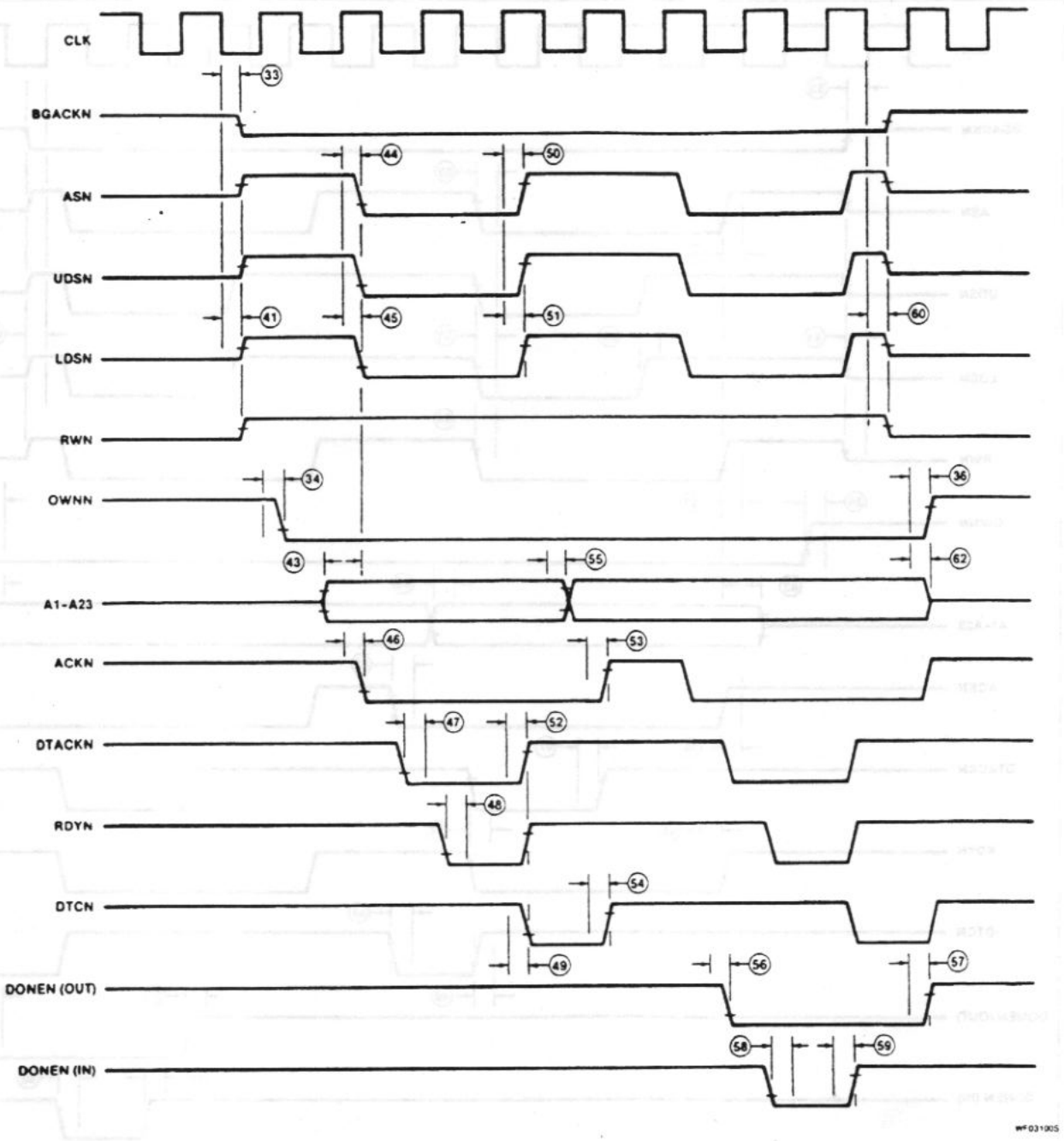
Figure 9. DMAI Burst Mode Request Timing

NOTES:
1. To maintain mastership next bus cycle, REQN must remain asserted at least until 1/2 clock cycle after DTCN is asserted.
2. To guarantee that mastership is relinquished next cycle, REQN must be negated no later than 1/2 clock cycle prior to DTCN.



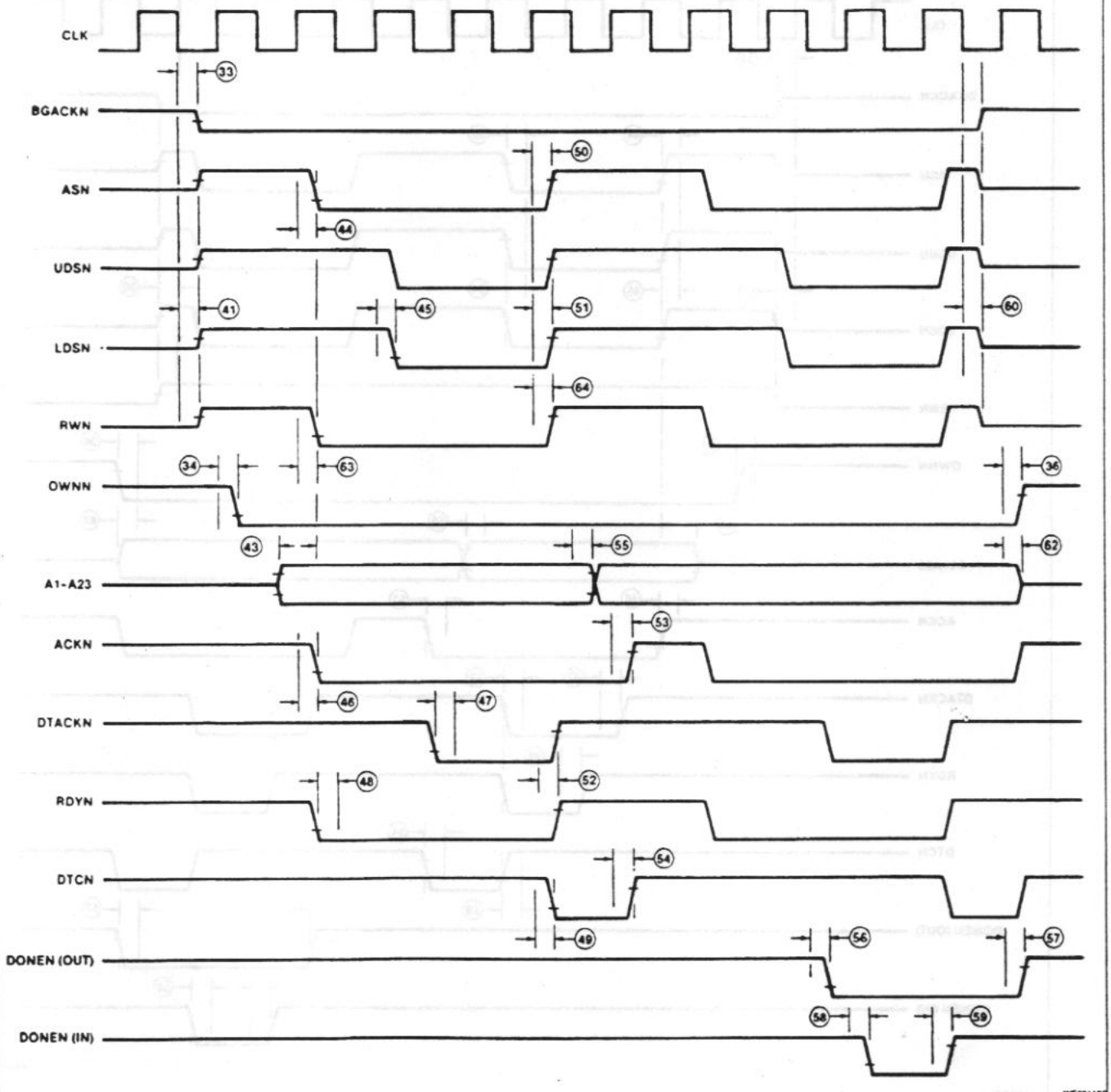
1. In order to keep the bus, REQN must come no later than the 1/2 clock minus the set-up time @ prior to assertion edge of DTCN.

Figure 10. DMAI Cycle Steal Mode Request Timing



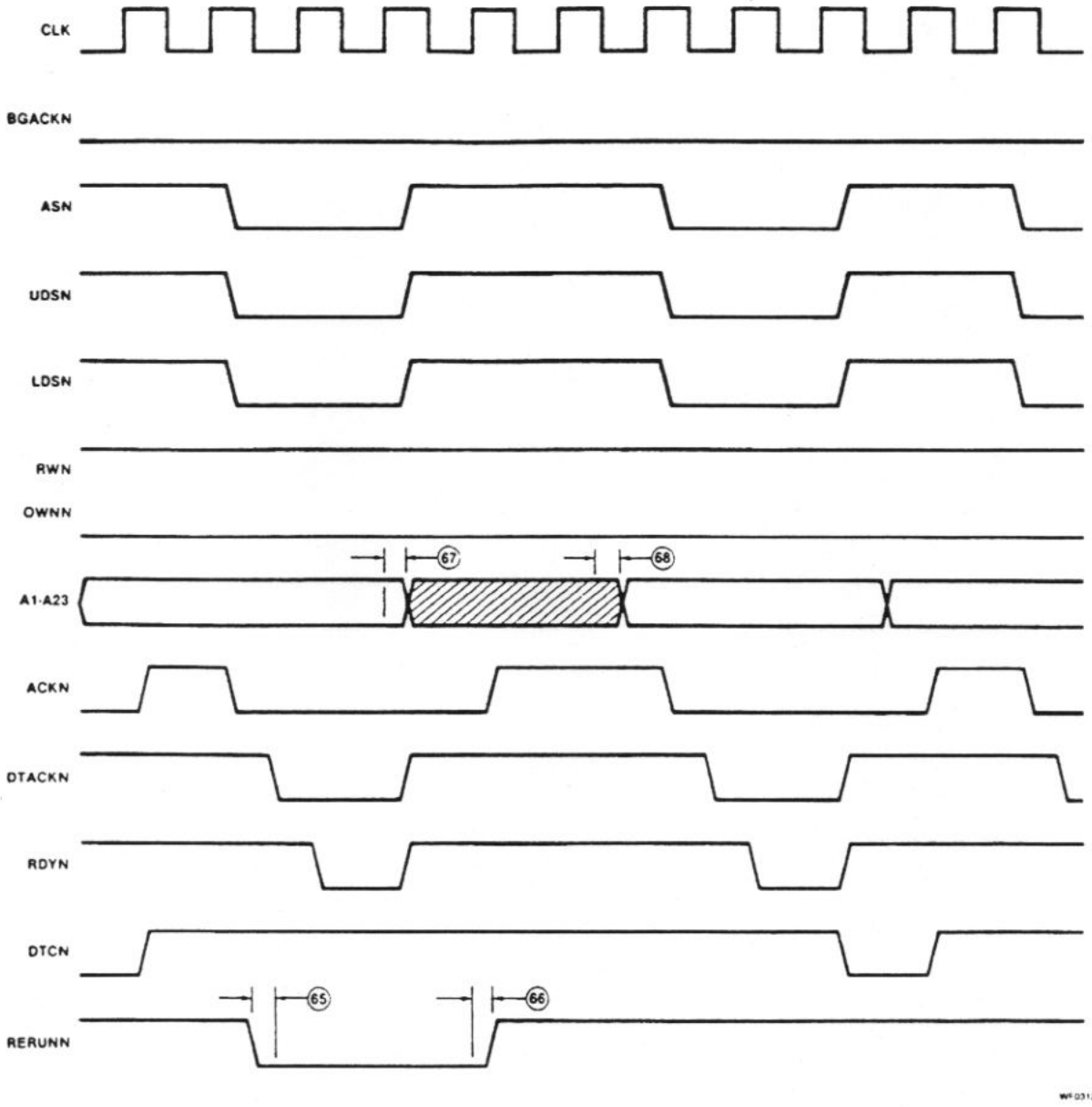
NOTE:
1. 16-bit transfer illustrated. For 8-bit transfer either LDSN or UDSN, but not both, will be asserted each cycle, depending on byte address.

Figure 11. Read from Memory, Write to Device



NOTE:
1. 16-bit transfer illustrated. For 8-bit transfer either LDSN or UDSN, but not both, will be asserted each cycle, depending on byte address.

Figure 12. Write to Memory, Read from Device



NOTES:
1. 16-bit transfer illustrated. For 8-bit transfer either LDSN or UDSN, but not both, will be asserted each cycle, depending on byte address.
2. DMAI will release the bus after a RERUNN if there is no valid request. The next request will then retry the cycle which was terminated by the RERUNN signal.
3. RERUNN must be asserted no later than DTACKN and RDYN.
4. If a cycle is terminated by RERUNN, the transfer count will be one less than the actual data transferred correctly. The double RERUNN signal on the same cycle will terminate the DMAI operation with a status bit set and an interrupt generated (if enabled).

Figure 13. Rerun Asserted During Read from Memory, Write to Device

APPENDIX C

DMA Test Software Listing

The following two software listings, DMABTM and DMABTS, are included as examples for test software that can be utilized for testing the VMIVME-DMA Boards in a back-to-back mode.

```
                        ********************************************************************************
                        ******              THIS PART OF THE TEST SOFTWARE IS                     ******
                        ******                 FOR THE MASTER'S CHASSIS                           ******
                        ********************************************************************************
                        ******  THIS DMA TEST TESTS 2 DMA'S IN BACK TO BACK MODE                 ******
                        ******  IN SEPERATE CHASSIS. THE TEST PREFORMS DMA TRANSFERS IN          ******
                        ******  BOTH DIRECTIONS IN CYCLE STEAL AND BURST MODES.  SET JUMPERS     ******
                        ******  AS SHOWN IN THE MANUAL FOR BUS REQUEST LEVEL 3 AND GRANT LEVEL  ******
                        ******  3.  CABLE P3 TO P3 AND P4 TO P4.  BOARD BASE                     ******
                        ******  ADDRESS IS SET FOR $E000. RUN THIS SOFTWARE IN CHASSÍS ONE.      ******
                        ******  CREATED APRIL 23, 1988 BY MGL. REV1                              ******
                        ******  REV2 JAN 11, 1988 CONTINOUS LOOP ADDED BY MGL                    ******
                        ********************************************************************************

          00001000              ORG.S   $1000
00001000 3E7C1F00               MOVE.W  #$1F00,A7           INITIALIZE STACK POINTER
00001004 4BFB125A               LEA     MS68,A5             SIGNON MESSAGE
00001008 4DFB129E               LEA     ENMS68,A6
0000100C 4E4F                   TRAP    #15
0000100E 0006                   DC.W    6

00001010 123C00B8       START   MOVE.B  #$B8,D1             SETUP D1 FOR TRANSFER IN CYCLE STEAL
00001014 4246                   CLR     D6

00001016 283C5A5A5A5A   START1  MOVE.L  #$5A5A5A5A,D4       FIRST DATA PATTERN TO BE TRANSFERED
0000101C 612A                   BSR.S   LDMEM1
0000101E 613A                   BSR.S   DMAOUT

00001020 283CA5A5A5A5           MOVE.L  #$A5A5A5A5,D4       SECOND DATA PATTERN
00001026 6120                   BSR.S   LDMEM1
00001028 6130                   BSR.S   DMAOUT

0000102A 7800                   MOVE.L  #$00000000,D4       THIRD  DATA PATTERN
0000102C 611A                   BSR.S   LDMEM1
0000102E 612A                   BSR.S   DMAOUT

00001030 78FF                   MOVE.L  #$FFFFFFFF,D4       FORTH DATA PATTERN
00001032 6114                   BSR.S   LDMEM1
00001034 6124                   BSR.S   DMAOUT
00001036 0C060008               CMP.B   #8,D6
0000103A 6708                   BEQ.S   DIN
0000103C 123C0028               MOVE.B  #$28,D1             SET TRANSFER FOR BURST MODE
00001040 4EFB1016               JMP.S   START1
00001044 4EFB10F8       DIN     JMP.S   START3

0000104B 41F900040000   LDMEM1  LEA     $40000,A0           LOAD MEM. WITH DATA PATTERN
0000104E 20C4           INITMEM1 MOVE.L D4,(A0)+
00001050 B1FC00041000           CMPA.L  #$41000,A0
00001056 66F6                   BNE     INITMEM1
00001058 4E75                   RTS

                        ********************************************************************************
                        *****TRANSFER DMA BLOCK OUT*****************************************************
                        ********************************************************************************


0000105A 46FC2700       DMAOUT  MOVE.W  #$2700,SR           MASK INTERUPTS
0000105E 163C0000               MOVE.B  #0,D3
```

```
 59  00001062 13FC004500FF          MOVE.B  #$45,$FFE04D        LOAD ATTN. INTERRUPT VECTOR
          E04D
 60  00001068A 13FC004000FF         MOVE.B  #$40,$FFE025        LOAD DMA INTERRUPT VECTOR
          E025
 61  00001072 13FC000B00FF          MOVE.B  #$B,$FFE007         ENABLE DMA INTERRUPTS
          E007
 62  0000107A 13C100FFE004          MOVE.B  D1,$FFE004          SET UP TRANSFER TYPE
 63  00001080 13FC00B000FF          MOVE.B  #$B0,$FFE000        CLEAR CSR
          E000
 64  00001088 13FC003900FF          MOVE.B  #$39,$FFE065        LOAD LWORD* AND ADDRESS MODIFIER
          E065
 65                      ********************************************************************************
 66                      ***********68000 TYPE CPU'S MULTIPLY THE INTERRUPT VECTOR BY 4*****************
 67                      ***********DURING AN INTERRUPT CYCLE.  EXAMPLE :  $45 *$4 = $114***************
 68                      ********************************************************************************
 69
 70  00001090 21FC000011D8          MOVE.L  #ATTN,$114          ATTN INT VECTOR LOCATION
          0114
 71  00001098 21FC000011AE          MOVE.L  #DONE1,$100         DMA DONE VECTOR LOCATION
          0100
 72  000010A0 21FC000011FA          MOVE.L  #I13,$104           DMA ERROR VECTOR TO LOCATION 113
          0104
 73                      ********************************************************************************
 74
 75
 76  000010A8 33FC040000FF          MOVE.W  #$400,$FFE00A       LOAD TRANSFER COUNT
          E00A
 77  000010B0 23FC00040000          MOVE.L  #$40000,$FFE00C     LOAD MEM ADD COUNTER
          00FFE00C
 78  000010BA 13FC003200FF          MOVE.B  #$32,$FFE005        LOAD DCR
          E005
 79  000010C2 13FC001E00FF          MOVE.B  #$1E,$FFE045        ATTN CONTROL REG. LEVEL 6 INT.
          E045
 80  000010CA 13FC000800FF          MOVE.B  #$0B,$FFE061        SET SPARE BIT TO DMA2 BOARD
          E061
 81  000010D2 267C00FFE060 WAIT     MOVE.L  #$FFE060,A3
 82  000010D8 3813                  MOVE.W  (A3),D4             WAIT FOR DMA2 TO RESPOND
 83  000010DA 02440080              AND.W   #$0080,D4           WITH SPARE BIT SET
 84  000010DE 0C440080              CMP.W   #$0080,D4
 85  000010E2 66EE                  BNE.S   WAIT
 86  000010E4 13FC001200FF          MOVE.B  #$12,$FFE061        ENABLE ATTN OUT,ATTN IN
          E061
 87  000010EC 46FC2000              MOVE.W  #$2000,SR
 88  000010F0 0C030001      SELF1   CMP.B   #1,D3               WAIT FOR SOMETHING TO HAPPEN
 89  000010F4 66FA                  BNE.S   SELF1
 90  000010F6 4E75                  RTS
 91
 92                      ********************************************************************************
 93                      *********RECIEVE DMA BLOCK*****************************************************
 94                      ********************************************************************************
 95
 96  000010FB 123C00B8      START3  MOVE.B  #$B8,D1             SET TRANSFER TO CYCLE STEAL
 97  000010FC 1C3C0000              MOVE.B  #0,D6
 98  00001100 13FC000000FF          MOVE.B  #0,$FFE063          FAIL LED ON
          E063
 99
100  00001108 283C5A5A5A5A  START2  MOVE.L  #$5A5A5A5A,D4       FIRST DATA PATTERN RECIEVED
101  0000110E 6122                  BSR.S   DMAIN
```

```
102
103  00001110 203CA5A5A5A5          MOVE.L   #$A5A5A5A5,D4        SECOND DATA PATTERN RECIEVED
104  00001116 611A                  BSR.S    DMAIN
105
106  00001118 7800                  MOVE.L   #$00000000,D4        THIRD DATA PATTERN RECIEVED
107  0000111A 6116                  BSR.S    DMAIN
108
109  0000111C 78FF                  MOVE.L   #$FFFFFFFF,D4        FORTH DATA PATTERN RECIEVED
110  0000111E 6112                  BSR.S    DMAIN
111  00001120 0C060008              CMP.B    #8,D6                COUNT THE NUMBER OF BLOCKS RECIEVED
112  00001124 6708                  BEQ.S    DOUT
113  00001126 123C0028              MOVE.B   #$28,D1              SET TRANSFER TO BURST MODE
114  0000112A 4EF81108              JMP.S    START2
115  0000112E 4EF81010     DOUT     JMP.S    START
116
117
118  00001132 46FC2700     DMAIN    MOVE.W   #$2700,SR            MASK INTERRUPTS
119  00001136 163C0008              MOVE.B   #8,D3
120  0000113A 33FC040000FF          MOVE.W   #$400,$FFE00A        LOAD TRANSFER COUNT
          E00A
121  00001142 23FC00041000          MOVE.L   #$41000,$FFE00C      LOAD MEM ADD COUNT
          00FFE00C
122  0000114C 13FC00B000FF          MOVE.B   #$B0,$FFE000         CLEAR CSR
          E000
123  00001154 13C100FFE004          MOVE.B   D1,$FFE004           SET TRANSFER TYPE
124  0000115A 13FC00B200FF          MOVE.B   #$B2,$FFE005         LOAD DCR
          E005
125  00001162 13FC004400FF          MOVE.B   #$44,$FFE04D         LOAD ATTN INT VECTOR
          E04D
126  0000116A 13FC004600FF          MOVE.B   #$46,$FFE025         LOAD DMA INT VECTOR
          E025
127  00001172 21FC000011BE          MOVE.L   #ATTN1,$110          ATTN VECTOR LOCATION
          0110
128  0000117A 21FC000011FA          MOVE.L   #I13,$11C            DMA ERROR
          011C
129  00001182 21FC0000121A          MOVE.L   #DONE2,$118          DMA DONE VECTOR
          0118
130  0000118A 13FC003900FF          MOVE.B   #$39,$FFE065         LOAD LWORD* ADDR MODIFIER
          E065
131  00001192 13FC001D00FF          MOVE.B   #$1D,$FFE045         ATTN CONTROL REG,LEVEL 5 INT.
          E045
132  0000119A 13FC001800FF          MOVE.B   #$18,$FFE061         ENABLE ATTN. INT
          E061
133  000011A2 46FC2000              MOVE.W   #$2000,SR
134  000011A6 0C030001     SELF2    CMP.B    #1,D3                WAIT FOR INTERRUPT
135  000011AA 66FA                  BNE.S    SELF2
136  000011AC 4E75                  RTS
137
138
139                        **************DONE1 INTERRUPT SERVICE ROUTINE**********************************
140  000011AE 5206         DONE1    ADD.B    #1,D6
141  000011B0 163C0001              MOVE.B   #1,D3
142  000011B4 13FC008000FF          MOVE.B   #$80,$FFE063
          E063
143  000011BC 4E73                  RTE
144
145                        **************ATTN1 INTERRUPT SERVICE ROUTINE**********************************
146  000011BE 13FC003D00FF ATTN1    MOVE.B   #$3D,$FFE047         ENABLE DMA INT
```

```
                E047
147   000011C6 13FC008800FF          MOVE.B  #$88,$FFE007          START DMA CONTROLER
                E007
148   000011CE 13FC000300FF          MOVE.B  #$03,$FFE061          ENABLE ATTN INT OUT, GO BIT SET, RECEIVE
                E061
149   000011D6 4E73                  RTE
150
151                          ************ATTN INTERRUPT SETVICE ROUTINE************************************
152
153   000011DB 13FC008000FF ATTN     MOVE.B  #$80,$FFE063          ATTN SERVICE ROUTINE
                E063
154   000011E0 13FC003D00FF          MOVE.B  #$3D,$FFE047          LOAD DICR (ENA DMA INTERRUPT)
                E047
155   000011E8 13FC008800FF          MOVE.B  #$88,$FFE007          START DMA CONTROLER
                E007
156   000011F0 13FC000500FF          MOVE.B  #$05,$FFE061          HIT GO BIT TO ENABLE HANDSHAKE, CYCLE
                E061
157   000011F8 4E73                  RTE
158
159
160   000011FA 4BF81205      I13     LEA.L   MSG3,A5               DMA ERROR SERVICE ROUTINE
161   000011FE 4DF81219              LEA.L   ENMSG3,A6
162   00001202 4E4F                  TRAP    #15
163   00001204 00                    DC.B    0
164
165   00001205 0D0A          MSG3    DC.B    $D,$A
166   00001207 4D4153544552          DC.B    'MASTER DMA ERROR'
167   00001217 0D0A                  DC.B    $D,$A
168   00001219               ENMSG3
169
170                          ********************************************************************************
171                          ******DONE2 INTERRUPT SERVICE ROUTINE ****************************************
172                          ********************************************************************************
173                          ***THIS SERVICE ROUTINE CHECKS THE RECIEVER BUFFER FOR DATA ERRORS*************
174                          ********************************************************************************
175
176   0000121A 5206          DONE2   ADD.B   #1,D6
177   0000121C 163C0001              MOVE.B  #1,D3
178   00001220 2B7C00041000          MOVE.L  #$41000,A4            STARTING ADDRESS OF RECIEVER BUFFER
179   00001226 303C0000              MOVE.W  #0,D0
180   0000122A 2A1C          NOSHIFT MOVE.L  (A4)+,D5
181   0000122C BA84                  CMP.L   D4,D5                 COMPARE BUFFER
182   0000122E 660A                  BNE.S   DATAERR
183   00001230 5240                  ADD.W   #1,D0
184   00001232 0C400400              CMP.W   #$400,D0
185   00001236 66F2                  BNE.S   NOSHIFT
186   00001238 4E73                  RTE
187
188   0000123A 4BF8124A      DATAERR LEA     DERR,A5
189   0000123E 4DF8125A              LEA     ENDERR,A6
190   00001242 4E4F                  TRAP    #15
191   00001244 0006                  DC.W    6
192   00001246 4E4F                  TRAP    #15
193   00001248 0000                  DC.W    0
194
195   0000124A 0D0A          DERR    DC.B    $D,$A                 SCREEN PRINTED MESSAGES
196   0000124C 444154412045          DC.B    'DATA ERROR'
197   00001256 0D0A                  DC.B    $D,$A
```

```
198   00001258 0D0A                DC.B    $D,$A
199   0000125A              ENDERR
200
201   0000125A 0D0A        MSG8     DC.B    $D,$A
202   0000125C 445231315720         DC.B    'DR11W TEST IS IN PROGRESS'
203   00001275 0D0A                 DC.B    $D,$A
204   00001277 464C41534849         DC.B    'FLASHING LEDS INDICATE A PASSING TEST'
205   0000129C 0D0A                 DC.B    $D,$A
206   0000129E              ENMSG8
207                                 END

****** TOTAL ERRORS      0--
****** TOTAL WARNINGS    0--
```

SYMBOL TABLE LISTING

SYMBOL NAME	SECT	VALUE	SYMBOL NAME	SECT	VALUE
ATTN		000011D8	I13		000011FA
ATTN1		000011BE	INITMEM1		0000104E
DATAERR		0000123A	LDMEM1		00001048
DERR		0000124A	MS63		00001205
DIN		00001044	MS68		0000125A
DMAIN		00001132	NOSHIFT		0000122A
DMAOUT		0000105A	SELF1		000010F0
DONE1		000011AE	SELF2		000011A6
DONE2		0000121A	START		00001010
DOUT		0000112E	START1		00001016
ENDERR		0000125A	START2		00001100
ENMS63		00001219	START3		000010FB
ENMS68		0000129E	WAIT		000010D2

```
                        ********************************************************************************
                        *****              THIS PART OF THE TEST SOFTWARE IS                       *****
                        *****                   FOR THE SLAVE'S CHASSIS                            *****
                        ********************************************************************************
                        ******   THIS DMA TEST TESTS 2 DMA'S IN BACK TO BACK MODE                   *****
                        ******   IN SEPERATE CHASSIS. THE TEST PREFORMS DMA TRANSFERS IN            *****
                        ******   BOTH DIRECTIONS IN CYCLE STEAL AND BURST MODES.  SET JUMPERS       *****
                        ******   AS SHOWN IN THE MANUAL FOR BUS REQUEST LEVEL 3 AND GRANT LEVEL     *****
                        ******   3. CABLE P3 TO P3 AND P4 TO P4.  BOARD BASE                        *****
                        ******   ADDRESS IS SET FOR $E000. RUN THIS SOFTWARE IN CHASSIS TWO.        *****
                        ******   CREATED FEB. 23, 1988 BY MGL. REV1                                *****
                        ********************************************************************************

           00001000             ORG.S   $1000
00001000 3E7C1F00               MOVE.W  #$1F00,A7           INITALIZE STACK POINTER
00001004 4BF8126C               LEA     MSG8,A5
00001008 4DF812AE               LEA     ENMSG8,A6           SIGN ON MESSAGE
0000100C 4E4F                   TRAP    #15
0000100E 0006                   DC.W    6

00001010 4245         RESTART   CLR     D5
00001012 13FC000000FF           MOVE.B  #$00,$FFE063
         E063
0000101A 1C3C0000               MOVE.B  #0,D6
0000101E 123C00B8               MOVE.B  #$B8,D1             SET TRANSFER TYPE TO CYCLE STEAL
                        ****************DMA IN**********************************************************
00001022 283C5A5A5A5A START     MOVE.L  #$5A5A5A5A,D4       FIRST DATA PATTERN RECIEVED
00001028 6122                   BSR.S   DMAIN

0000102A 283CA5A5A5A5           MOVE.L  #$A5A5A5A5,D4       SECOND DATA PATTERN RECIEVED
00001030 611A                   BSR.S   DMAIN

00001032 7800                   MOVE.L  #$00000000,D4       THIRD DATA PATTERN RECIEVED
00001034 6116                   BSR.S   DMAIN

00001036 78FF                   MOVE.L  #$FFFFFFFF,D4       FORTH DATA PATTERN RECIEVED
00001038 6112                   BSR.S   DMAIN

0000103A 0C060008               CMP.B   #8,D6               CHECK THE NUMBER OF BLOCKS RECIEVED
0000103E 6708                   BEQ.S   DOUT
00001040 123C0028               MOVE.B  #$28,D1             SET TRANSFER TYPE TO BURST MODE
00001044 4EF81022               JMP.S   START
00001048 4EF81104     DOUT      JMP.S   START1



0000104C 163C0000     DMAIN     MOVE.B  #0,D3
00001050 46FC2700               MOVE.W  #$2700,SR           MASK INTERRUPTS
00001054 33FC040000FF           MOVE.W  #$400,$FFE00A       LOAD TRANSFER COUNT
         E00A
0000105C 23FC00041000           MOVE.L  #$41000,$FFE00C     LOAD MEM ADD COUNT
         00FFE00C
00001066 13FC00B000FF           MOVE.B  #$B0,$FFE000        CLEAR CSR
         E000
0000106E 13C100FFE004           MOVE.B  D1,$FFE004          SET TRANSFER TYPE
00001074 13FC00B200FF           MOVE.B  #$B2,$FFE005        LOAD DCR
         E005
```

```
54  0000107C 13FC004400FF          MOVE.B  #$44,$FFE04D        LOAD ATTN INT VECTOR
             E04D
55  00001084 13FC004000FF          MOVE.B  #$40,$FFE025        LOAD DMA INT VECTOR
             E025
56  0000108C 21FC00000108          MOVE.L  #ATTN1,$110         ATTN VECTOR LOCATION
             0110
57  00001094 21FC00000122C         MOVE.L  #I13,$104           DMA ERROR
             0104
58  0000109C 21FC00000102          MOVE.L  #DONE1,$100         DMA DONE VECTOR
             0100
59  000010A4 13FC00390FF           MOVE.B  #$39,$FFE065        LOAD LWORD* ADDR MODIFIER
             E065
60  000010AC 13FC001D00FF          MOVE.B  #$1D,$FFE045        ATTN CONTROL REG,LEVEL 5 INT.
             E045
61  000010B4 13FC001800FF          MOVE.B  #$18,$FFE061        SPARE, ATTN ENABLE
             E061
62  000010BC 46FC2000              MOVE.W  #$2000,SR
63  000010C0 0C030001      SELF2   CMP.B   #1,D3               WAIT FOR INTERRUPT
64  000010C4 66FA                  BNE.S   SELF2
65  000010C6 4E75                  RTS
66                         **************ATTN1 INTERRUPT SERVICE ROUTINE**********************************
67  000010C8 13FC003D00FF  ATTN1   MOVE.B  #$3D,$FFE047        ENABLE DMA INT
             E047
68  000010D0 13FC000800FF          MOVE.B  #$08,$FFE007        START DMA CONTROLER
             E007
69  000010DB 13FC000300FF          MOVE.B  #$03,$FFE061        ENABLE ATTN INT OUT, GO BIT SET, RECEIVE
             E061
70  000010E0 4E73                  RTE
71
72                         ********************************************************************************
73                         **************DONE1 DMA INTERRUPT SERVICE ROUTINE*****************************
74                         *****THIS ROUTINE CHECKS THE RECIEVER BUFFER FOR DATA ERRORS********************
75                         ********************************************************************************
76
77  000010E2 5206          DONE1   ADD.B   #1,D6
78  000010E4 163C0001              MOVE.B  #1,D3
79  000010E8 287C00041000          MOVE.L  #$41000,A4          BEGINNING ADDRESS OF DATA BLOCK
80  000010EE 303C0000              MOVE.W  #0,D0
81  000010F2 2A1C          NOSHIFT MOVE.L  (A4)+,D5            CHECK DATA
82  000010F4 BAB4                  CMP.L   D4,D5
83  000010F6 66000154              BNE.L   DATAERR
84  000010FA 5240                  ADD.W   #1,D0
85  000010FC 0C400400              CMP.W   #$400,D0
86  00001100 66F0                  BNE.S   NOSHIFT
87  00001102 4E73                  RTE
88
89                         ************DMA OUT*************************************************
90  00001104 163C0000      START1  MOVE.B  #0,D3
91  00001108 1C3C0000              MOVE.B  #0,D6
92  0000110C 123C00B8              MOVE.B  #$B8,D1             SET TRANSFER TYPE TO CYCLE STEAL
93  00001110 13FC00B000FF          MOVE.B  #$B0,$FFE063
             E063
94
95  00001118 283C5A5A5A5A START2   MOVE.L  #$5A5A5A5A,D4       FIRST DATA BLOCK TRANSFERED
96  0000111E 612A                  BSR.S   LDMEM1
97  00001120 613A                  BSR.S   DMAOUT
98
99  00001122 283CA5A5A5A5          MOVE.L  #$A5A5A5A5,D4       SECOND DATA BLOCK TRANSFERD
```

```
100  00001128 6120              BSR.S   LDMEM1
101  0000112A 6130              BSR.S   DMAOUT
102
103  0000112C 7800              MOVE.L  #$00000000,D4       THIRD DATA BLOCK TRANSFERED
104  0000112E 611A              BSR.S   LDMEM1
105  00001130 612A              BSR.S   DMAOUT
106
107  00001132 78FF              MOVE.L  #$FFFFFFFF,D4       FORTH DATA BLOCK TRANSFERED
108  00001134 6114              BSR.S   LDMEM1
109  00001136 6124              BSR.S   DMAOUT
110
111  00001138 BC060008          CMP.B   #8,D6               CHECK NUMBER OF BLOCKS TRANSFERED
112  0000113C 6708              BEQ.S   DIN
113  0000113E 123C0028          MOVE.B  #$28,D1             SET TRANSFER TYPE TO BURST MODE
114  00001142 4EF81118          JMP.S   START2
115  00001146 4EF81010    DIN   JMP.S   RESTART
116
117  0000114A 41F900040000 LDMEM1 LEA    $40000,A0          LOAD DATA BLOCK
118  00001150 20C4         INITMEM1 MOVE.L D4,(A0)+
119  00001152 B1FC00041000      CMPA.L  #$41000,A0
120  00001158 66F6              BNE.S   INITMEM1
121  0000115A 4E75              RTS
122
123  0000115C 46FC2700    DMAOUT MOVE.W  #$2700,SR         MASK INTERUPTS
124  00001160 163C0000          MOVE.B  #0,D3
125  00001164 13FC004500FF      MOVE.B  #$45,$FFE04D        LOAD ATTN. INTERRUPT VECTOR
          E04D
126  0000116C 13FC004600FF      MOVE.B  #$46,$FFE025        LOAD DMA INTERRUPT VECTOR
          E025
127  00001174 13FC000B00FF      MOVE.B  #$B,$FFE007         ENABLE DMA INTERRUPTS
          E007
128  0000117C 13C100FFE004      MOVE.B  D1,$FFE004          SET UP TRANSFER TYPE
129  00001182 13FC00B000FF      MOVE.B  #$B0,$FFE000        CLEAR CSR
          E000
130  0000118A 13FC003900FF      MOVE.B  #$39,$FFE065        LOAD LWORD* AND ADDRESS MODIFIER
          E065
131                     ********************************************************************************
132                     ***********68000 TYPE CPU'S MULTIPLY THE INTERRUPT VECTOR BY 4*****************
133                     ***********DURING AN INTERRUPT CYCLE.  EXAMPLE :  $45 *$4 = $114****************
134                     ********************************************************************************
135
136  00001192 21FC000011FA      MOVE.L  #ATTN,$114          ATTN INT VECTOR LOCATION
          0114
137  0000119A 21FC0000121C      MOVE.L  #DONE,$118          DMA DONE VECTOR LOCATION
          0118
138  000011A2 21FC0000122C      MOVE.L  #I13,$11C           DMA ERROR VECTOR TO LOCATION 113
          011C
139                     ********************************************************************************
140
141
142  000011AA 33FC040000FF      MOVE.W  #$400,$FFE00A       LOAD TRANSFER COUNT
          E00A
143  000011B2 23FC00040000      MOVE.L  #$40000,$FFE00C     LOAD MEM ADD COUNTER
          00FFE00C
144  000011BC 13FC003200FF      MOVE.B  #$32,$FFE005        LOAD OCR
          E005
145  000011C4 13FC001E00FF      MOVE.B  #$1E,$FFE045        ATTN CONTROL REG. LEVEL 6 INT.
          E045
```

```
146  000011CC 13FC000900FF          MOVE.B   #$0B,$FFE061        SET FNCT3 BIT TO DR11W2 BOARD
              E061
147  000011D4 267C00FFE060 WAIT     MOVE.L   #$FFE060,A3
148  000011DA 3813                  MOVE.W   (A3),D4             WAIT FOR DR11W2 TO RESPOND
149  000011DC 02440080              AND.W    #$0080,D4           WITH FNCT3 BIT SET
150  000011E0 0C440080              CMP.W    #$0080,D4
151  000011E4 66EE                  BNE.S    WAIT
152  000011E6 13FC001200FF          MOVE.B   #$12,$FFE061        ENABLE ATTN OUT,ATTN IN
              E061
153  000011EE 46FC2000              MOVE.W   #$2000,SR
154  000011F2 0C030001      SELF1   CMP.B    #1,D3               WAIT FOR SOMETHING TO HAPPEN
155  000011F6 66FA                  BNE.S    SELF1
156  000011F8 4E75                  RTS
157
158
159
160                         ************ATTN INTERRUPT SETVICE ROUTINE**************************************
161
162  000011FA 13FC00B800FF ATTN     MOVE.B   #$B0,$FFE063        ATTN SERVICE ROUTINE
              E063
163  00001202 13FC003D00FF          MOVE.B   #$3D,$FFE047        LOAD DICR (ENA DMA INTERRUPT)
              E047
164  0000120A 13FC00B800FF          MOVE.B   #$B8,$FFE007        START DMA CONTROLER
              E007
165  00001212 13FC000500FF          MOVE.B   #$05,$FFE061        HIT GO BIT TO ENABLE HANDSHAKE
              E061
166  0000121A 4E73                  RTE
167
168
169
170                         *****DMA DONE INTERRUPT SERVICE ROUTINE*****************************************
171
172  0000121C 5206          DONE    ADD.B    #1,D6
173  0000121E 163C0001              MOVE.B   #1,D3
174  00001222 13FC000800FF          MOVE.B   #$80,$FFE063        TURN LED OFF
              E063
175  0000122A 4E73                  RTE
176
177
178  0000122C 4BF81237      I13     LEA      MSG3,A5             DMA ERROR ROUTINE
179  00001230 4DF8124B              LEA      ENMSG3,A6
180  00001234 4E4F                  TRAP     #15
181  00001236 00                    DC.B     0
182
183  00001237 0D0A          MSG3    DC.B     $D,$A
184  00001239 4D4153544552          DC.B     'MASTER DMA ERROR'
185  00001249 0D0A                  DC.B     $D,$A
186  0000124B               ENMSG3
187
188
189  0000124C 4BF8125C      DATAERR LEA.L    DERR,A5
190  00001250 4DF8126C              LEA.L    ENDERR,A6
191  00001254 4E4F                  TRAP     #15
192  00001256 0006                  DC.W     6
193  00001258 4E4F                  TRAP     #15
194  0000125A 0000                  DC.W     0
195
196  0000125C 0D0A          DERR    DC.B     $D,$A
```

```
197   0000125E 444154412045          DC.B    'DATA ERROR'
198   00001268 0D0A                  DC.B    $D,$A
199   0000126A 0D0A                  DC.B    $D,$A
200   0000126C              ENDERR
201
202   0000126C 0D0A         MSG8     DC.B    $D,$A
203   0000126E 444D41205445          DC.B    'DMA TEST IS IN PROGRESS'
204   00001285 0D0A                  DC.B    $D,$A
205   00001287 464C41534849          DC.B    'FLASHING LEDS INDICATE A PASSING TEST'
206   000012AC 0D0A                  DC.B    $D,$A
207   000012AE              ENMSG8
208
209                                  END

****** TOTAL ERRORS      0--
****** TOTAL WARNINGS    0--
```

SYMBOL TABLE LISTING

SYMBOL NAME	SECT	VALUE	SYMBOL NAME	SECT	VALUE
ATTN		000011FA	I13		0000122C
ATTN1		000010C8	INITMEM1		00001150
DATAERR		0000124C	LDMEM1		0000114A
DERR		0000125C	MSG3		00001237
DIN		00001146	MSG8		0000126C
DMAIN		0000104C	NOSHIFT		000010F2
DMAOUT		0000115C	RESTART		00001010
DONE		0000121C	SELF1		000011F2
DONE1		000010E2	SELF2		000010C0
DOUT		00001048	START		00001022
ENDERR		0000126C	START1		00001104
ENMSG3		0000124B	START2		00001118
ENMSG8		000012AE	WAIT		000011D4

DOCUMENTATION EVALUATION FORM

VMIC welcomes your comments and suggestions.

Please return this form to:

VME MICROSYSTEMS INTERNATIONAL CORPORATION
12090 South Memorial Parkway
Huntsville, Alabama 35803-3308
(205) 880-0444
1-800-322-3616

Evaluation: Please rate the following areas on a scale of 1 to 5 (1 = Poor; 5 = Excellent).

DOCUMENT NO.: _____
REVISION DATE: _____

READABILITY	_____	ILLUSTRATIONS	_____
ORGANIZATION	_____	PROGRAMMING INFORMATION	_____
ACCURACY	_____	SPECIFICATIONS	_____
COMPLETENESS	_____	MAINTENANCE DIAGRAMS	_____

SPECIFIC PROBLEMS: **PAGE(s)**

( ) CLARIFICATION REQUIRED _____

( ) NOT ENOUGH INFORMATION GIVEN _____

( ) TYPOGRAPHICAL ERRORS _____

( ) TECHNICAL ERRORS (EXPLAIN): _____

DOCUMENT USE: (check all that apply)

( ) HARDWARE	( ) SOFTWARE	( ) PRODUCT EVALUATION
( ) OPERATION	( ) MAINTENANCE	( ) TRAINING

ADDITIONAL COMMENTS: _____

YOUR NAME: _____
TITLE: _____
COMPANY: _____
MAIL STOP: _____
STREET: _____
CITY, STATE, ZIP: _____
PHONE: _____

VMIC
047

Artisan Technology Group is an independent supplier of quality pre-owned equipment

Gold-standard solutions

Extend the life of your critical industrial, commercial, and military systems with our superior service and support.

We buy equipment

Planning to upgrade your current equipment? Have surplus equipment taking up shelf space? We'll give it a new home.

Learn more!

Visit us at **artisantg.com** for more info on price quotes, drivers, technical specifications, manuals, and documentation.

Artisan Scientific Corporation dba Artisan Technology Group is not an affiliate, representative, or authorized distributor for any manufacturer listed herein.

We're here to make your life easier. How can we help you today?

(217) 352-9330 | sales@artisantg.com | artisantg.com

ARTISAN® TECHNOLOGY GROUP