VI. 16-BIT MICROPROCESSOR PODS

WARNINGS

BEFORE PLUGGING IN THE ML4400 TO AN AC POWER SOURCE, VERIFY THAT THE CORRECT VOLTAGE (115 OR 230) HAS BEEN SELECTED VIA THE VOLTAGE SELECTOR SWITCH AT THE UPPER RIGHT OF THE REAR PANEL.

DO NOT INSERT OR REMOVE ANY CAPTURE MODULE, AND DO NOT CONNECT OR DISCONNECT ANY LOGIC POD OR MICROPROCESSOR POD FROM THE ML4400, WITHOUT FIRST POWERING DOWN THE ML4400 VIA THE POWER SWITCH AT THE LOWER RIGHT OF THE REAR PANEL. FAILURE TO DO SO MAY DAMAGE THE UNIT.

A. GENERAL

1. CONNECTIONS AND ADAPTOR POD

See Sections II-E and II-F, "Standard Capture Module and Microprocessor Pods" and "Microprocessor Pod and Microprocessor", regarding connections.

Note that two 16-Bit Microprocessor Pods, the 16M-680 (for the Motorola 68000/10) and the 16I-086 (for the Intel 8086/88) require an Adaptor for ML4100 Microprocessor Pods (AD-4100) when connected to an ML4400. One Adaptor pod is needed for each Microprocessor Pod being used simultaneously (up to a maximum of four).

2. DISASSEMBLY

See Section IV-D.4, "Disassembly Display Mode", as well as various parts of this section about the particular Microprocessor Pod being used.

3. CONFIGURATION AND SETUP

When the ML4400 is configured with a Microprocessor Pod instead of a Logic Pod, there are several differences in its Setup screens, as discussed below.

a. Format Display Screen (via Format key)

The ML4400 will automatically configure itself at powerup for synchronous microprocessor cycle data collection. The default State format has four fields defined:

EXT, a binary field for the external probes
STS, status line (usually)
ADR, address bus
DAT, data bus

The Data Inputs are labeled on the Format Display screen as probe numbers. For specific probe assignments for a particular 16-bit Microprocessor Pod, see the manual section for that Pod (Section VI-B and following, below).

b. Clock Setup Screen (via CLOCK key)

The external clock signal is derived as required from the signals on each microprocessor. (External clock qualification is not used with Microprocessor Pods.)

(An internal clock mode is not available with Microprocessor Pods.)

c. Trigger Words Setup Screen (via TRIGGER key)

When a Microprocessor Pod is used with the ML4400, two additional softkeys are available on the Trigger Words Setup screen:

The ROLL STATUS softkey speeds up editing the Status fields by rolling through common choices for Status values (fetch, read, write, etc.). Status mnemonics are displayed to the right of the Status field.

The STATUS BIT DEF'S softkey opens the Status Bit help window, which defines each bit in the Status field. Depressing the softkey a second time closes the window. For specific status values for a particular 16-bit Microprocessor Pod, see the manual section for that Pod (Sections VI-B and following, below).

d. Status, State, and State Search Screens

In these screens, status mnemonics are displayed to the right of each Status field.

4. OPERATION

For operating details for each of Arium's 16-Bit Microprocessor Pods, see Sections VI-B and following. (There is one section per Pod.)

B. 16M-680 MICROPROCESSOR POD (Supports Motorola 68000/68010)

1. GENERAL

The 16M-680 Microprocessor Pod clips directly over a 68000 or 68010 DIP. The pod collects data at microprocessor clock speeds of up to 16 MHz, and formats it for logic analysis and full program disassembly (including all exceptions and bus cycles) by the ML4400.

The pod has three operating modes (Filtered, Transparent, and Jump), and produces disassembled code in each of them. (The six Pod modes are designated onscreen by 68000F, 68000T, 68000JMP, 68010F, 68010T, and 68010JMP.) Special hardware in the pod tracks every bus cycle of the microprocessor and determines if the particular data will be recorded in the pod's current operating mode. In the Filtered mode, all unused prefetch cycles are removed in the pod hardware so they won't activate unwanted trigger events and won't be stored in the ML4400 memory. In the Transparent mode, all bus cycles are passed along to the ML4400. In the Jump mode, all bus cycles are filtered out except for changes in program flow (branches, subroutine calls, exceptions, etc.), providing a disassembled "map" of how the code was actually executed.

The Disassembly display details all microprocessor data cycles ("unscrambled" and put back together with their corresponding instructions), provides labels for readability, and marks all instructions as either "u" for User or "s" for Supervisor, depending upon how they are fetched. The State display shows data as it occurs on the address and data busses. Data may also be viewed on the Timing display; however, there is no internal clocking capability, and only in the Transparent or Jump modes are samples stored in strictly increasing time order. (Filtered mode delays the storage of instruction cycles relative to that of data cycles.)

If the microprocessor ceases operation (e.g., because of a HALT condition or a lack of DTACK, BERR, or VPA acknowledge during a bus cycle), several preceding bus cycles will remain stored in the pod. These cycles cannot be recorded or triggered upon until the microprocessor resumes operation.

2. HARDWARE

a. Switch Settings

On one end of the 16M-680 pod there is an access hole for a four-position DIP switch. This access hole is on the left end of the pod when viewed with the label lettering upright.

Switch 1 selects the processor type:

Down = 68000

Up = 68010 (To select the 68010, Switch 3 must also be Up.)

NOTE: ML4400 power must be cycled (turned off, then on again) when ever the position of Switch 1 has been changed.

Switch 2 selects the data source for Probes 30, 29, and 28 (the leftmost three significant bits of the EXT field on the State display):

Down = the Interrupt Priority Level lines on the processor (IPL2*, IPL1*, & IPL0*, active low, are Probes 30, 29, and 28, respectively.)

Up = the external clip leads

NOTE: The position of this switch may be changed during use; i.e., the power needn't be cycled (turned off, then on again).

Switch 3 selects the filtering type:

Down = 68000 systems without wait states or DMA cycles

Up = 68010, & 68000 systems with wait states or DMA cycles

The effect of this switch is complicated and is explained in Section VI-B.4, "Filtered Mode", below.

NOTE: The position of this switch may be changed during use; i.e., the power needn't be cycled (turned off, then on again).

(Switch 4 is unused in this pod.)

b. Signal Assignments

Fifty-four signals are saved in each cycle of the trace buffer:

16 Data lines	
24 Address lines	(The least significant address line, A0, is created by hardware in the pod.)
14 Discrete lines	(One of them is used for test and has no meaning to the user)

The data is saved in the trace buffer inverted, so the sense of the data displayed is automatically set to INV on the Format screen. The probes are displayed in four groups (Address, Data, Status and External.) Figures IV-A and IV-B, below, relate probe numbers to signals and describe the signals.

Figure VI-A

SIGNAL ASSIGNMENTS FOR 16M-680 POD

(Asterisk indicates active low)

PROBE NO.	68000/68010 SIGNAL
External (EXT) -- (Field of 7 binary numbers)	
30	IPL2*, or Lead 02 if external signals are used
29	IPL1*, or Lead 01 if external signals are used
28	IPL0*, or Lead 00 if external signals are used
27	HALT*
26	(Used for test)
25	VMA*
24	BERR*
Status (STS) -- (Field of 7 binary numbers)	
70	FETCH* or SOURCE, depending on Filtered or Jump mode (generated by the Pod)
69	BGACK*
68	BYTE
67	R/W*
66	FC2
65	FC1
64	FC0
Address (ADR) -- (Field of 6 hexadecimal numbers)	
63-60	A23-A20
59-56	A19-A16
55-52	A15-A12
51-48	A11-A08
47-44	A07-A04
43-40	A03-A00
Data (DAT) -- (Field of 4 hexadecimal numbers)	
15-12	D15-D12
11-8	D11-D08
7-4	D07-D04
3-0	D03-D00

Figure VI-B

68000/68010 EXTERNAL AND STATUS SIGNALS
FOR 16M-680 POD

(Asterisk indicates active low)

Probe 71: FETCH* is created by the pod in the Filtered mode; it is asserted during the first cycle of an instruction.

Probe 70: SOURCE is created by the pod in the Jump mode. It is high if the cycle is a Source, and low if it is a Destination; if the cycle is both a Destination and a Source, the bit will be low. If in Filtered mode, this bit indicates the first word of an instruction fetch.

Probe 69: BGACK* is the Bus Grant Acknowledge input to the processor. When low, cycles are performed by a device other than the processor. In Filtered mode, such cycles are removed from the data stream; in Transparent mode, any BGACK* cycles in the buffer are ignored by the disassembler.

Probe 68: BYTE is created by pod hardware; it is high when only one data strobe is asserted.

Probe 67: R/W* is the READ/WRITE* signal from the processor; it is high during Read cycles and low during Write cycles.

Probes 66-64: FC2, FC1, and FC0 are the processor function codes which indicate which of several address spaces is accessed by a given cycle. (These "spaces" are listed Section 3, below, "Status Codes.")

Probe 32: A0 is created by pod hardware; it is high when the lower data strobe (LDS) is asserted and the upper data strobe (UDS) is negated.

Probes 30-28: Lead 02, Lead 01, & Lead 00, or IPL2*, IPL1*, & IPL0* (Interrupt Priority Level lines), depending upon whether the position of Switch 2 on the pod is Up or Down, respectively. (See Sec. B.1, above, "Switch Settings.")

Probe 27: HALT* -- See Probe 24, BERR*.

Probe 26: Used for test, and undefined for normal operation.

Probe 25: VMA* is the Valid Memory Address output from the processor, asserted in response to the VPA* (Valid Peripheral Address) signal. These signals are used primarily when interfacing with 6800 peripherals. If VPA* is asserted by external hardware during an Interrupt Acknowledge cycle, the processor performs an autovector operation. Note that data sampling is different during VMA* cycles, using the falling edge of E rather than the rising edge of the data strobes. (See microprocessor data sheet for details.)

Probes 24, 27: BERR* and HALT* signals are taken directly off the processor. When the BERR* signal is asserted without HALT*, the processor takes a bus error exception; when BERR* and HALT* are both asserted, the processor will retry the cycle. In the Filtered mode, cycles with both BERR* and HALT* asserted (cycles to be retried) are removed, as the cycle will recur.

c. Status Codes

The EXT (External) and STS (Status) fields (on the default Trigger screen and State Display screen) contain status information about the bus cycle. The three least significant bits of the STS fields are the function codes FC2, FC1 and FC0 of the processor, which define the "space" of the cycle as shown:

FC2	FC1	FC0	
0	0	1	User Data space
0	1	0	User Program space
1	0	1	Supervisor Data space
1	1	0	Supervisor Program space
1	1	1	CPU space; used for Interrupt Acknowledge and Breakpoint cycles

(Other combinations are unassigned and cannot be accessed in the 68000; the 68010 can access them via the MOVES instruction.)

Figure VI-C

TRIGGER SETUP SCREEN WITH 16M-680 POD

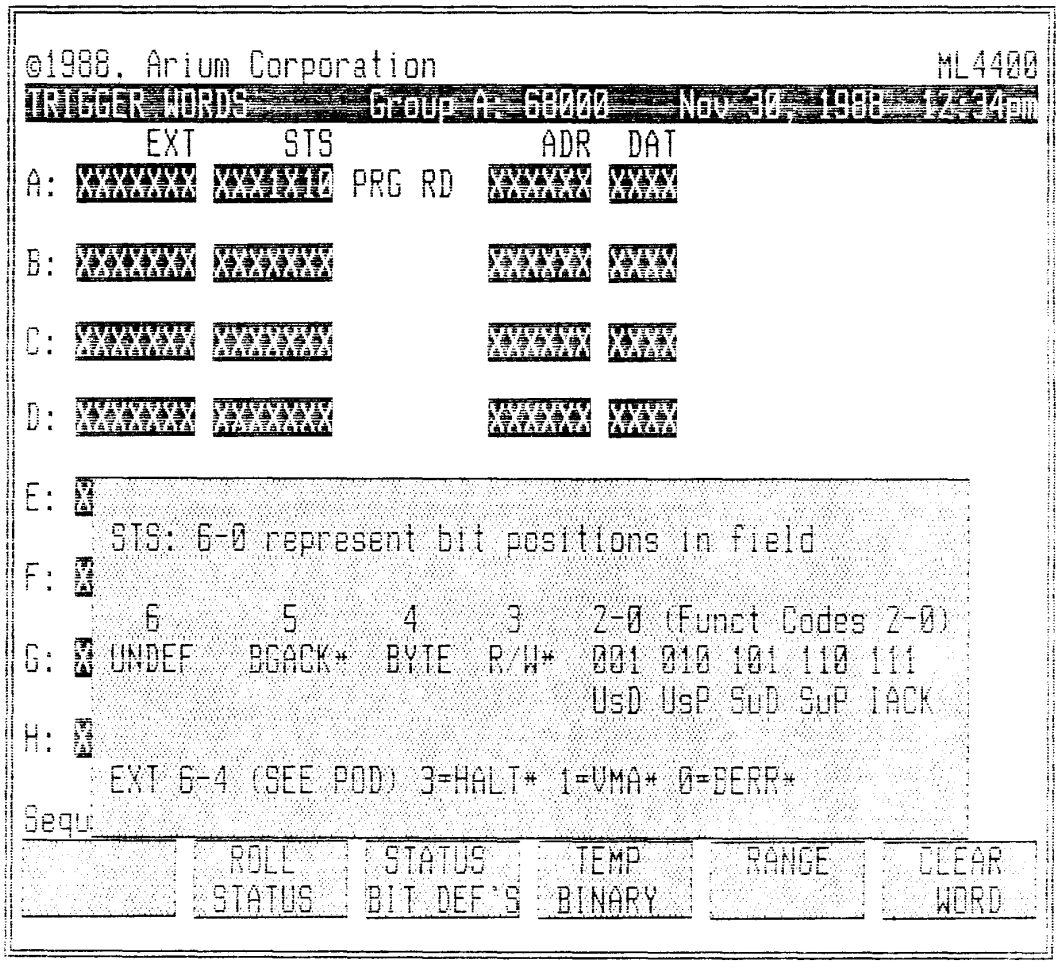


Figure VI-D

68000/68010 STATUS CODE EXAMPLES
FOR 16M-680 POD

	STS
Read User Data Space Word	X101001
Write User Data Space Word	X100001
Read User Data Space Byte	X111001
Read User Program Space Word	X101010
Read User Program Space First Instruction Word (Filtered mode only)	0101010
Read Supervisor Program Space First Instruction Word	0101110
Write Supervisor Data Space Byte	X110101
Interrupt Acknowledge (The level of the interrupt being acknowledged is available on Address Lines A3, A2, and A1. Also, in the 68010, Interrupt Acknowledges must be distinguished from breakpoints by using upper address lines.)	X101111

	EXT	STS
Autovectored Interrupt Acknowledge (VMA* asserted in response to VPA*)	XXXXX0X	X101111
Bus Error Cycle	XXX1XX0	X1XXXXX
Retry Cycle (Transparent mode only)	XXX0XX0	X1XXXXX
VMA* Cycle (special hardware)	XXXXX0X	
Nonprocessor Cycle (DMA)	XXXXXXX	X0XXXXX

---

d. Probe Color Codes for External Input Lines

Probe No.	Lead No.	Input Wire	Twisted with	Tip Color
28	0	Beige	Brown (inactive)	Black
29	1	Red	Beige (inactive)	Brown
30	2	Orange	Beige (inactive)	Red

3. TRANSPARENT MODE

In the Transparent mode, all bus cycles (including unused prefetches) are sent to the ML4400 logic analyzer in the order in which they occur. Data captured in the Transparent mode can be viewed in all three display formats: State, Timing and Disassembly. The disassembler discards unused prefetches, so that disassembled data is the same as it is in the Filtered mode.

(See also Sections VI-B.6, "Triggering Considerations", and VI-B.4.b.(2), "FIFO Memory.")

4. FILTERED MODE

a. General

As with most 16-bit microprocessors, the 68000 uses an instruction prefetch mechanism to enhance performance. While one instruction is being executed, the two words following it have already been fetched from program memory. If an instruction or exception causes a branch, one or both of the prefetched words will be discarded. With logic analyzers that fail to filter prefetched words, triggering on a location in program memory guarantees only that the instruction has been fetched, not executed.

To eliminate undesired triggering, the Arium 16M-680 Microprocessor Pod tracks every bus cycle and filters out unused prefetches from the instruction stream sent to the ML4400 Logic Analyzer. Increased data storage is an added benefit of filtering, because unused prefetches do not take up any space in the 500-cycle trace buffer of the ML4400.

Data captured in the Filtered mode may be displayed in State, Timing and Disassembly formats.

b. Hardware Considerations

(1) General

When in Filtered mode, special hardware in the 16M-680 Microprocessor Pod monitors the processor's bus cycles to determine when prefetched cycles have not been executed. Because as many as thirty bus cycles may occur between the read of an instruction and its execution (or its being discarded), the pod must temporarily hold the instruction off to one side of the data stream until it has enough information to determine whether the cycle has actually been executed.

The result is that (in Filtered mode) all instruction words appear in the ML4400 trace buffer and in the State display where the next instruction word would have appeared if in Transparent mode, except for instruction words which are found to be unused prefetches, which are removed completely.

Note that, due to the delay of Filtered mode, the cycles appear in the ML4400 memory in the same order in which the processor actually performed them, not in the order in which they were fetched off of the bus by the prefetch mechanism. This may be important to the user when viewing external signals or interrupt levels on the State display.

(2) FIFO Memory

In Filtered mode, a small (4-cycle), first-in-first-out memory in the pod removes cycles in the event of an interrupt, when two prefetched cycles are discarded. If the processor stops execution, this FIFO memory will contain the last four cycles and they will not have been captured in the ML4400 trace buffer. (In Transparent mode, the length of this FIFO memory is reduced to one cycle.)

(3) Unremoved Prefetches

Three instructions (CHK, DIV, and TRAPV) and the TRACE exception generate exceptions in a way that makes it difficult to remove all prefetches; in fact, it is sometimes impossible to know whether one or two prefetches need to be removed. In these cases only one prefetched word (the last) is removed.

Address and Bus Error exceptions do not have unused prefetches, but a cycle may be deleted anyway. The address of this cycle is pushed on the stack and labeled "AA" (see Section G.3.c.(4), "Exceptions"), so the loss of this cycle should not pose a problem.

Another ambiguous circumstance occurs when the processor executes a single-word conditional branch around a one-word instruction that makes no data accesses. The difficulty arises because the data stream is identical, regardless of whether or not the branch is taken. In the 68010, the execution speed is also identical, regardless of path, and so it is not possible to know whether the one-word instruction has been executed. The 68000 takes two clock cycles longer when the instruction has been executed than when it has not been executed, and the pod monitors this in order to decide whether or not the instruction has been skipped and is to be discarded. The clock cycle counting information may not be valid in systems with wait states or in which DMA cycles can occur. In either case, the one-word instruction following the branch is marked with a "?" in the disassembly display to indicate the uncertainty about its actual execution.

The usage of the cycle counting information is controlled by the user, via Switch 3 on the pod. For 68010 systems, this switch must be Up, or all such instructions will be removed. For most 68000 systems, this switch should be Down. For 68000 systems with DMA or wait states, the switch should usually be Up; the switch may be Down, but the user must be aware of the possibility of cycles being filtered out which have actually been executed.

(4) Self-Synchronization

In order to make intelligent decisions about unused prefetches, the pod must synchronize itself with the processor. Under ordinary circumstances, only a few cycles are required for synchronization, and it will actually self-synchronize before the ML4400 begins recording data.

There are certain patterns of instructions for which the pod is not guaranteed to synchronize itself, but such patterns are virtually always tight loops of fewer than ten unusual instructions with no Writes. If such a pattern occurs, the pod may incorrectly remove cycles from the data stream, and the user must either use the Transparent mode, or restart data acquisition in the Filtered mode in hopes that the pod will synchronize itself. (Such self-synchronization may indeed occur, because synchronization may depend upon the exact point in the pattern at which acquisition begins.)

A processor RESET will cause the pod to lose sync. It should resynchronize within a few cycles, but for analyzing the processor's behavior at RESET, Transparent mode is recommended.

5. JUMP MODE

The Jump mode filters out all bus cycles, except those preceding and following events that cause changes in program flow (such as branches, subroutine calls, and exceptions). This feature is very useful for checking program flow and determining which sections of code are being executed. Data captured in the Jump mode can be viewed in all three display formats: State, Timing and Disassembly.

The Disassembly display screen describes each entry as a Source ("was branched from") or Destination ("was branched to"). Since only the first word of each instruction is stored in the buffer, only the mnemonic of each instruction is displayed. If an instruction is both a Destination and a Source (e.g., when a jump lands at another jump), the instruction appears only as a Destination in the state buffer (i.e., no source entry is recorded, and there will be two Destination entries in a row). Thus, it is always possible to set a trigger on a Destination, but those Source instructions which are also Destinations will never be recorded with the "SOURCE" bit set in the STS field.

6. TRIGGERING CONSIDERATIONS

Undesired Triggering on Prefetches in Transparent Mode: In the Transparent mode, prefetched instructions that are not executed may cause undesired triggering, i.e., triggering of the logic analyzer without the occurrence of the event of interest. Such unused prefetches are found before interrupts, after branches and returns, and sometimes after jumps and calls. Because these instructions have not been executed, they will not appear on the Disassembly screen.

To avoid undesired triggering, use the Filtered mode, which strips unused prefetches from the data stream (described in Section VI-B.4, "Filtered Mode").

7. DATA DISPLAY

a. General

With the 16M-680 Pod in Transparent mode, the State display shows all instruction fetches and data cycles in the same order in which they occurred on the bus. In Filtered mode, the order of cycles in the trace buffer will not match the actual order of cycles on the processor bus (see Section VI-B.4, "Filtered Mode"), and any unused fetches (fetches following an instruction causing a program discontinuity) are disqualified by the Pod and not stored in the trace buffer.

The Disassembly display groups these program fetches and data cycles into 68000 assembly language instructions. Regardless of whether or not the Pod is filtering unused fetches, these unused fetches are not displayed on the Disassembly screen.

Figure VI-E

STATE DISPLAY SCREEN
WITH 16M-680 POD (ALL MODES)

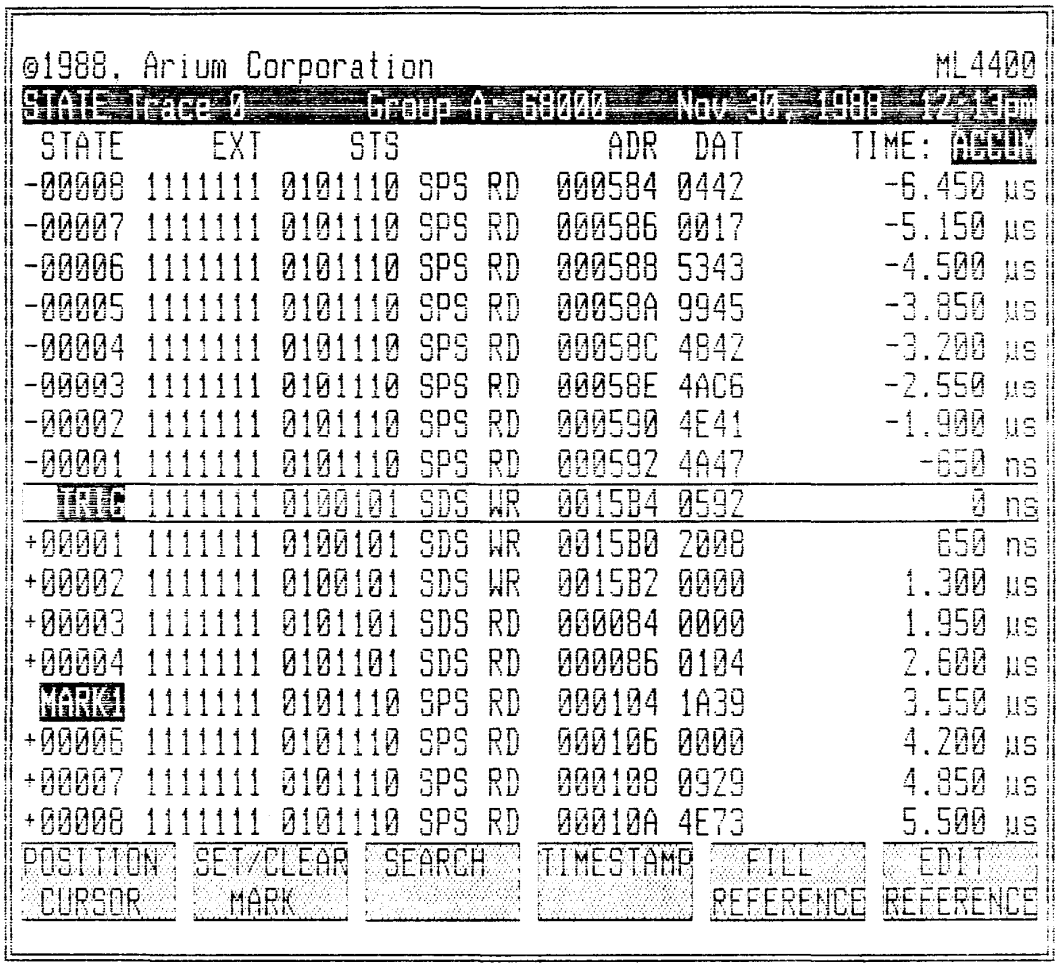


Figure VI-F

DISASSEMBLY DISPLAY SCREEN
WITH 16M-680 POD
(TRANSPARENT AND FILTERED MODES)

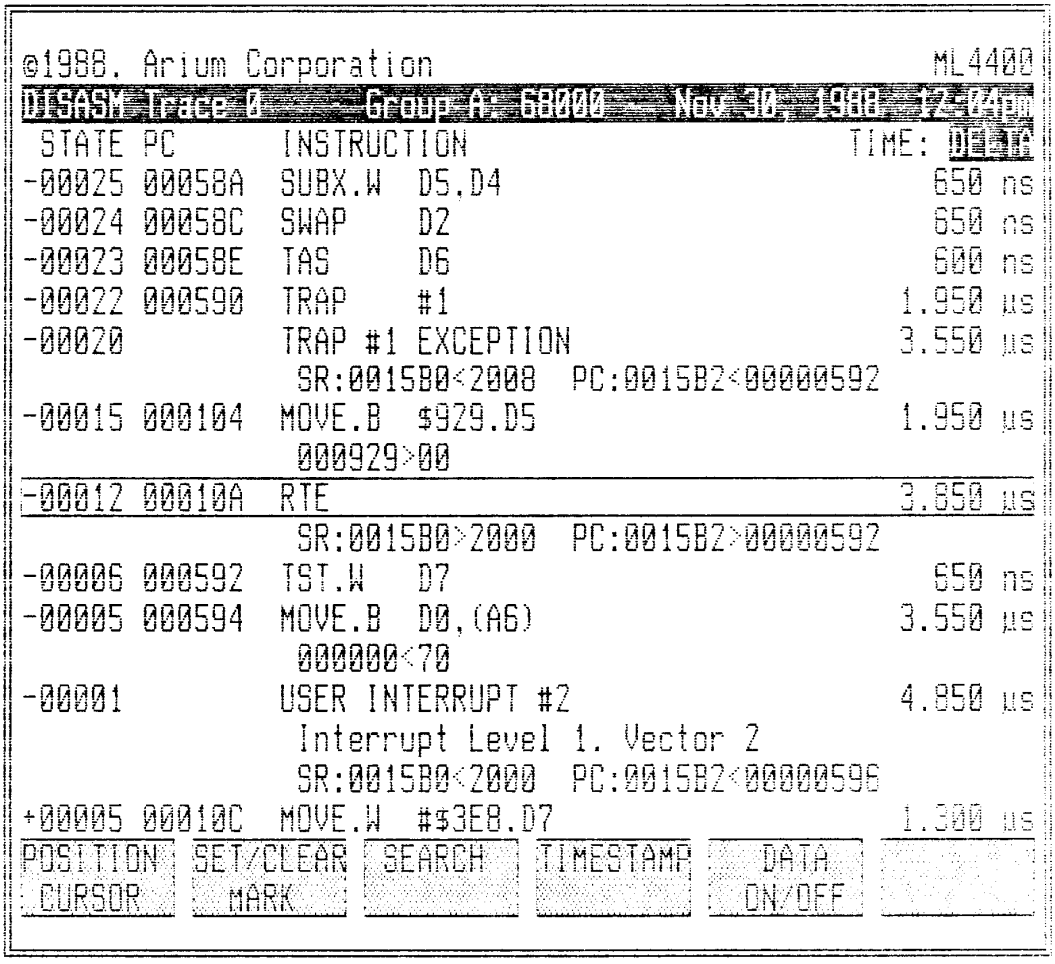


Figure VI-G

DISASSEMBLY DISPLAY SCREEN
WITH 16M-680 POD
(JUMP MODE)

(Not yet available)

The formatting and display of data is described generally for the ML4400 in Section IV, "Operation." Further details of data display when using 16-bit microprocessor pods, including typical State and Disassembly display screens, keyboard functions, and display formats, follow.

b. Display Formats

(1) Instruction Display

As with 8-bit microprocessor pod disassembly, each instruction displayed contains a state number corresponding to the instruction position relative to the trigger event, the value of the Program Counter when the instruction was fetched (PC), and the mnemonic and operands for the instruction itself. Also, with the 68000/68010 pod, an "s" or "u" at the far right of the screen indicates whether the instruction was fetched from Supervisor space or from User space.

(2) Data Cycles Display

Data cycles are displayed on lines following the instruction that caused them. These cycles may be turned on or off by toggling the DATA CYCLES softkey.

Each data cycle contains an address, an indication of whether data was read from (>) or written to (<) the address, and the data itself. Each address consists of six hexadecimal characters (24 bits), while data may be a byte (2 characters), a word (4 characters), or a long word (8 characters).

Even though the 68000 performs a long word access by doing two word accesses, the ML4400 concatenates the data and shows it as as one data cycle on the Disassembly screen.

(3) Specific Instruction Display Techniques

(a) MOVEM

The register list coded with the MOVEM instruction is reconstructed and shown as the second line of the instruction. This line is part of the instruction and stays displayed even with data cycles off. The data cycles associated with the MOVEM instruction are in the same order (reading left to right) as they are on the register list. See Figure VI-F, "Disassembly Display Screen with 16M-680 Pod (Transparent and Filtered Modes)".

(b) BSR, JSR

The data cycle showing the value of the Program Counter pushed onto the stack is labelled "PC:".

(c) RTD, RTS, RTR

The data cycle showing the value of the Program Counter popped from the stack is labelled "PC:".

(d) Exceptions

PC, SR, VO, IR, AA Labels

The data cycles showing the values of the Program Counter and Status Register pushed onto the stack are labelled "PC:" and "SR:", respectively. If the processor is a 68010, the data cycle showing the Vector Offset is labelled "VO:". On a 68000 bus error or address error exception, the data cycle showing the Instruction Register push is labelled "IR:" and the cycle showing the Access Address push is labelled "AA:".

68010 Bus or Address Error Exception

On a 68010 bus error or address error exception, the following data cycles are labelled:

FA:	Fault Address
DO:	Data Output Buffer
DI:	Data Input Buffer
II:	Instruction Input Buffer

Unlabelled Cycles

Unlabelled cycles correspond to Motorola internal information. (See Motorola M68000 Programmer's Reference Manual, M68000UM-(A)D4, for additional information.)

Interrupts

If the exception is an interrupt, then the type of interrupt (externally vectored, autovector, spurious or uninitialized), interrupt priority level, and interrupt vector number are displayed and labelled as such.

Exception Vector

All exceptions label the exception vector which is read as "VECTOR:".

8. DISASSEMBLY SOFTWARE

The Disassembly software displays only executed (as opposed to prefetched) instructions, regardless of whether the pod is in Filtered mode (sending up only executed cycles) or Transparent mode (sending up all cycles).

To do this, the disassembly software must "sync up" to the cycles in the trace buffer, in much the same way as the pod hardware syncs up to the processor in Filtered mode. The disassembly software finds the first program space read, assumes it is an opcode fetch, and checks the following cycles to see if it made a correct assumption. If the following cycles don't match the expected pattern of instruction fetches, data cycles and prefetches for that particular opcode value, then the software assumes it found a subsequent byte of an instruction or a program space operand read. It ignores this cycle, finds the next program space read, and repeats the process.

This software process of syncing up to the trace cycles is performed in both Transparent and Filtered modes. This seems unnecessary in Filtered mode, because the pod sends up information identifying the first word of an instruction. Data qualification, however, may filter out subsequent cycles of an instruction, so that relying upon the fetch bit may be insufficient.

If the software cannot make sense of the cycles in the trace buffer, it will display the message "Can't disassemble--PS reads used as opcodes". It then shows disassembly of the buffer, assuming that every PS read was the first word of an instruction. Each line of display is marked with a "?" to indicate the software's uncertainty with the disassembed code.

If the software cannot make sense out of some of the cycles in the buffer, but then can resync to the trace cycles, it displays the message "Unable to disassemble XXX cycles," where "XXX" is the number of processor cycles that did not fit the expected pattern.

In addition, a special case exists when a one-word instruction with no data cycles follows a Bcc instruction. The processor cycle pattern appears identical, whether or not the branch has actually been taken. The software cannot determine whether or not the instruction has been executed, so it marks it with a "?".

C. 16I-086 MICROPROCESSOR POD (Fully supports the Intel 8086/8087/8088)

NOTE: The 16I-086 partially supports the Intel 8089 coprocessor by collecting data, but without disassembly.

(If the Pod is to be clipped directly onto an 8087 or 8089, a hardware jumper inside the Pod must be moved; see Section VI-C.2.d, "Pod Connection Error Warning.")

1. GENERAL

The 16I-086 Microprocessor Pod clips directly over an 8086 or 8088 DIP. The pod collects data at microprocessor clock speeds of up to 12.5 MHz, and formats it for logic analysis and full program disassembly (including all exceptions and bus cycles) by the ML4400.

The 8086 and 8088 microprocessors have two operating modes, Minimum and Maximum. In Minimum mode, the processors provide few indications of their internal operation; in particular, no indication of instruction queue activity is available outside the processor. For Minimum-mode processors, the 16I-086 Microprocessor Pod has only one mode of operation--Transparent.

In Maximum mode, these processors provide queue status information for use by coprocessors, and the 16I-086 pod provides three operating modes of its own: Transparent, Filtered, and Jump. The Filtered and Jump modes use the queue status information to record only specific processor cycles.

There are a total of eight combinations of processor modes (Minimum and Maximum) with pod modes (Transparent, Filtered, and Jump), and the pod produces disassembled code in each of them. In the Transparent mode, all buscycles are passed along to the ML4400. In Filtered and Jump modes, special hardware in the pod tracks every bus cycle of the microprocessor and determines if the particular data will be recorded in the pod's current operating mode. In the Filtered mode, all unused prefetch cycles are removed in the pod hardware so they won't activate unwanted trigger events and won't be stored in the ML4400 memory. In the Jump mode, all bus cycles are filtered out except for changes in program flow (jumps, subroutine calls, interrupts, etc.), resulting in a disassembled "map" of where code was actually executed.

The ML4400 Disassembly display screen details all microprocessor data cycles "unscrambled" and put back together with their corresponding instructions. The State display screen shows data in the order in which it was transmitted by the pod. Data may also be viewed on the Timing display screen; however, there is no internal clocking capability, and only in the Transparent or Jump modes are samples stored in increasing time order. (Filtered mode delays storage of instruction cycles relative to that of data cycles.)

2. HARDWARE

a. Switch Settings

On one end of the 16I-086 pod, there is an access hole for a 4-position DIP switch. This access hole is on the left end of the pod when viewed with the label lettering upright.

Switch 1 selects the processor mode:

Down = MAX (Maximum)
Up = MIN (Minimum)

NOTE: ML4400 power must be cycled (turned off, then on again) whenever the position of this switch has been changed.

Switch 2 selects the data bus width (processor type):

Down = 8 bits (8088 processor)
Up = 16 bits (8086 processor)

NOTE: ML4400 power must be cycled (turned off, then on again) whenever the position of this switch has been changed.

Switch 3 controls instruction byte repacking, and applies to the use of the Filtered pod mode on 16-bit processors (8086 F) only:

Down = Repack instruction bytes into words before recording (unless a data cycle occurs on the processor bus after the low instruction byte is used, and before the high instruction byte is used).

Up = Send instruction bytes to the ML4400 as soon as they are used; make no specific attempt to repack the bytes. (Repacking may occur occasionally.)

NOTE: Switch 3 normally must be Down, or the disassembler may be unable to disassemble portions of the recorded data. This switch may be changed during use; i.e., the ML4400 power need not be cycled after resetting the the switch. (See also Sections VI-5.b and VI-7.)

Switch 4 controls Jump mode operation (Max.-mode processors only):

Down = Jump mode records only program flow discontinuities due to the INTR interrupt.

Up = Jump mode records program flow discontinuities due to all causes (jumps, calls, returns, software and hardware interrupts, and instruction traps).

NOTE: This switch may be changed during use; i.e, the ML4400 power need not be cycled after resetting the switch. The setting of this switch should be examined (and changed if necessary) whenever the user chooses to operate the pod in Jump mode. (See also Section VI-6.)

b. Signal Assignments

The signal assignments vary with the pod mode, which is dependent upon processor type (8086 or 8088) and processor mode (Minimum or Maximum). The following table lists the 8 combinations of processor modes and pod modes, and shows how they are designated on the ML4400 display screens.

Figure VI-H

DESIGNATIONS OF PROCESSOR/POD MODES
FOR 16I-086 POD

Modes Desig.	Processor Mode	Data Bus Width	Pod Recording Mode
8086 MIN	Minimum	16 bits	Transparent
8086 T	Maximum	16 bits	Transparent
8086 F	Maximum	16 bits	Filtered
8086 JMP	Maximum	16 bits	Jump
8088 MIN	Minimum	8 bits	Transparent
8088 T	Maximum	8 bits	Transparent
8088 F	Maximum	8 bits	Filtered
8088 JMP	Maximum	8 bits	Jump

Signal class assignments for the probe lines are given in Figures VI-I and VI-J, below.

Figure VI-I

8086/8088 SIGNAL ASSIGNMENTS (OVERVIEW)
FOR 16I-086 POD

External: All Modes
Probe 18 = Ext. Lead 1
Probe 17 = Ext. Lead 0
Probe 16 = Reset Flag

Status: Min. Processor Modes: Probes 68-60; Max. Processor Modes: Probes 70-60

Address: All Modes: Probes 59-40 (32 = LSB)

Data: 8086: Probes 15-0 (0 = LSB); 8088: Probes 7-0 (0 = LSB)

Figure VI-J

8086/8088 STATUS SIGNAL ASSIGNMENTS (DETAILED)
FOR 16I-086 POD

(Asterisk denotes active low)

Probe No.	Bit Position in STS Field	8086 MIN	8086 T	8086 F	8086 JMP
60	B0	BHE*	BHE*	BHE*	BHE*
61	B1	PHLD	LOCK*	LOCK*	LOCK*
62	B2	IAK*	S0*	S0*	S0*
63	B3	DT/R*	S1*	S1*	S1*
64	B4	M/IO*	S2*	S2*	S2*
65	B5	S3	S3	S3	S3
66	B6	S4	S4	S4	S4
67	B7	S5	S5	S5	S5
68	B8	S6(=0)	S6	S6	S6
69	B9	--	(always 1)	FL*	ID*
70	B10	--	(always 1)	FH*	SRC

Probe No.	Bit Position in STS Field	8088 MIN	8088 T	8088 F	8088 JMP
60	B0	BHE*	BHE*(=1)	BHE*(=1)	BHE*(=1)
61	B1	PHLD	LOCK*	LOCK*	LOCK*
62	B2	SS0*	S0*	S0*	S0*
63	B3	DT/R*	S1*	S1*	S1*
64	B4	IO/M*	S2*	S2*	S2*
65	B5	S3	S3	S3	S3
66	B6	S4	S4	S4	S4
67	B7	S5	S5	S5	S5
68	B8	S6(=0)	S6	S6	S6
69	B9	--	(always 1)	FL*	ID*
70	B10	--	(always 1)	(always 1)	SRC

Most status signals are sampled directly at the microprocessor, and their names correspond directly to 8086 or 8088 signals. Those status signals whose names do not correspond to 8086 or 8088 signals are shown in Figure VI-K, below.

Figure VI-K

16I-086 POD STATUS SIGNALS
WHOSE NAMES DO NOT CORRESPOND
TO 8086/8088 SIGNALS

PHLD is high (1) for the first recorded microprocessor-caused bus cycle following an occurrence of HLDA. This signal is present in Minimum mode only, and indicates that a DMA cycle (or cycles) probably occurred just prior to this microprocessor cycle. DMA cycles are not recorded in the trace buffer for Minimum-mode processors.

IAK* is low (0) for INTA* cycles, and is an approximate substitute for SO*, which is not available on the Minimum-mode 8086.

FL* is low (0) if the lower data byte is the first byte of an instruction.

FH* is low (0) if the upper data byte is the first byte of an instruction.

SRC is high (1), for 8086 JMP and 8088 JMP modes, if the indicated byte on the data bus is the Source of a discontinuity, and low (0) if the byte is the Destination of a discontinuity.

ID* is low (0), for 8086 JMP and 8088 JMP modes, if this Destination entry in the trace buffer is due to a discontinuity caused by an INTR hardware interrupt.

Figure VI-L

EXTERNAL FIELD SIGNAL ASSIGNMENTS
FOR 16I-086 POD

Probe No.	Signal	Function
16	Reset Flag	A latched signal set on the first bus cycle following reset (fetch at 0FFFF0H)
17	Ext. Lead 0	Samples about 12 ns after the processor data bus is sampled (at the end of each bus cycle)
18	Ext. Lead 1	Samples about 12 ns after the processor data bus is sampled (at the end of each bus cycle)

c. Status Codes

The STS field (on the Trigger setup screen and the State display screen) specifies status information about processor bus cycles. The signals on Probes 64-62 (usually processor signals S2*, S1*, and S0*) specify the purpose of the cycle (I/O read, I/O write, memory read, memory write, fetch, or interrupt acknowledge). The following table shows status codes for the various types of cycles.

Figure VI-M

TYPICAL 8086/8088 STATUS CODES
FOR TRIGGERING ON SPECIFIC CYCLE TYPES
FOR 16I-086 POD

	8086 and 8088 Maximum Modes	8088 MIN	8086 MIN
Interrupt Acknowledge	XXXXXX000XX	XXXX100XX	XXXX000XX
I/O Read	XXXXXX001XX	XXXX101XX	XXXX001XX
I/O Write	XXXXXX010XX	XXXX110XX	XXXX011XX
Fetch	XXXXXX100XX	XXXX000XX	XXXX101XX
Memory Read	XXXXXX101XX	XXXX001XX	
Memory Write	XXXXXX110XX	XXXX010XX	XXXX111XX

The signals on Probes 66 and 65 (S4 and S3) indicate which segment of memory was accessed by the bus cycle. The following examples specify memory segments in addition to cycle type.

Figure VI-N

TYPICAL 8086/8088 STATUS CODES
FOR TRIGGERING ON SPECIFIC MEMORY SEGMENT ACCESSES
FOR 16I-086 POD

	All Max. Mode Processors	8088 MIN	8086 MIN
Read from ES:	XXXX00101XX	XX00001XX	XX00101XX
Read from SS:	XXXX01101XX	XX01001XX	XX01101XX
Read from CS:	XXXX10101XX	XX10001XX	XX10101XX
Read from DS:	XXXX11101XX	XX11001XX	XX11101XX
Write to ES:	XXXX00110XX	XX00010XX	XX00111XX
Write to SS:	XXXX01110XX	XX01010XX	XX01111XX
Write to CS:	XXXX10110XX	XX10010XX	XX10111XX
Write to DS:	XXXX11110XX	XX11010XX	XX11111XX

By Intel convention, the signals on Probes 68 and 67 (S6 and S5) indicate the following:

If S6 = 0, then the microprocessor caused this bus cycle.

If S5 = 0, then INTR interrupts are disabled.
If S5 = 1, then INTR is enabled.

For Maximum-mode processors only:

If S6 = 1, then a coprocessor or other DMA device caused this bus cycle.

If S5 = 0, then the coprocessor is an 8087.
If S5 = 1, then the coprocessor is an 8089.

Note than when disassembly of 8087 instructions is enabled (by the DATA CYCLES softkey), or when DMA cycles are present, the disassembler relies upon the above conventions to distinguish 8086/8088 cycles from all others.

d. Probe Color Codes for External Input Lines

Probe No.	Lead No.	Input Wire	Twisted with	Tip Color
17	0	Beige	Brown (inactive)	Black
18	1	Red	Beige (inactive)	Brown

e. Pod Connection Error Warning

A "Pod Connection Error" warning is given by any of the ML4400 microprocessor pods if the pod is not properly connected to the target microprocessor, or if power is not applied to the target microprocessor. The 16I-086 pod is no exception, but this pod also issues the warning for another reason.

If the DIP switch settings indicate that the microprocessor is running in Minimum mode, but the MIN/MAX pin of the target microprocessor indicates Maximum mode (or vice versa), a "Pod Connection Error" will be issued. To correct this condition, the DIP switch settings must be changed and ML4400 power must be cycled.

The 8087 and 8089 coprocessors do not have a MIN/MAX pin, and the user may defeat the warning hardware to allow the pod to be clipped directly to one of these coprocessors. To defeat the warning, open the pod box and find JP2 (next to U42), then move the shorting jumper to short JP2 Pin 2 (middle pin) to JP2 Pin 3 (toward U49). (The shorting jumper is normally installed from JP2 Pin 1 to JP2 Pin 2.)

3. MINIMUM PROCESSOR MODE, TRANSPARENT MODE ONLY (8086 MIN and 8088 MIN)

The 8086 and 8088 microprocessors operate in two modes, Minimum and Maximum. Processors operating in Minimum mode provide few indications of their internal operation; in particular, there is no indication of instruction queue activity outside the processor. For Minimum-mode processors, then, the ML4400 offers only the Transparent pod mode, which causes the 8I-086 pod to transmit processor bus cycles to the ML4400 trace buffer in the same order in which they occur.

Unused prefetches (those instructions prefetched but later discarded due to a jump, interrupt or other discontinuity in program flow) may trigger the analyzer in spite of the fact that these instructions were not executed. Triggering on instructions which are occasionally unused prefetches can usually be overcome by triggering on a later instruction in the same code path (one which is further downstream from a jump, call, return, etc., which is causing the unused prefetch), or by using a more complicated trigger sequence to determine when the instruction is actually executed.

4. MAXIMUM PROCESSOR MODE WITH TRANSPARENT POD MODE (8086 T and 8088 T)

Because processors operating in Maximum mode provide queue status information, the 16I-086 microprocessor pod offers three "POD MODE" entries on the Format setup screen when the Maximum processor mode has been selected (via the pod DIP switch). The first of the three modes, Transparent (8086T or 8088T), does not use the queue status information; this pod mode causes the pod to send processor bus cycles as they occur. Unused prefetches are not stripped, and triggering on them may occur. If the unused prefetches cause triggering problems, put the pod in Filtered mode.

5. MAXIMUM PROCESSOR MODE WITH FILTERED POD MODE (8086 F and 8088 F)

a. 8086 F AND 8088 F Modes

In the Filtered modes (8086 F or 8088 F), the 16I-086 pod maintains an instruction queue of its own, and transmits the instruction bytes from the queue only when the queue status pins on the processor indicate that these bytes are actually used. Unused prefetches are never sent to the ML4400, and they cannot cause triggering. Instruction bytes are delayed in the pod until their actual execution; they will appear later in the trace buffer relative to nearby data (noninstruction) bus cycles. This slight delay of the instruction bytes has no effect upon the display of the trace buffer as disassembled code. The delay, and the absence of unused prefetch, relative to Transparent mode, can be seen only in the State data display.

In Filtered mode, the instruction queue is handled on a byte-by-byte basis. For the 8086, which prefetches words (two bytes) at a time, the pod has the capabiity of repacking the executed instruction bytes into the original words, and (via a DIP switch on the pod) the user can somewhat control the repacking of program bytes. The remainder of this section discusses the action of this switch, and applies only to the 8086 F mode.

b. 8086 F Mode; Repacking Instruction Bytes (See Section VI-C.7.c)

In the 8086 F mode only, Switch 3 in the Down position causes the pod to delay sending a lower byte of an instruction word until the disposition of the upper byte is known, or until a noninstruction data cycle occurs. (The pod will never allow a noninstruction bus cycle to enter the trace buffer ahead of an executed instruction byte which may have caused the data cycle.) So, whenever possible, the pod will repack a low-byte/high-byte pair of instruction bytes when Switch 3 is Down. This repacking will not occur if one of the bytes is not executed, or if a noninstruction bus cycle occurs between the execution of the two bytes. Keeping Switch 3 down uses the available trace buffer much more efficiently.

In the Up position, Switch 3 commands the pod, in Filtered mode, to transmit executed instruction bytes from its queue as soon as they are available. Bytes will be repacked into words only if, due to some delay such as the transmission of data cycles, the pod finds that a low/high pair of instruction bytes has been executed and neither has been transmitted to the ML4400. Thus, with Switch 3 up, repacking will occur only as required; the pod makes no special effort to repack the instruction bytes. The Up position is useful when the processor "hangs" after executing a WAIT or HALT. The pod will empty itself of all executed instructions rather than keep the last byte (if it was a low byte), waiting for another instruction.

Switch 3 may be changed at any time, even while the pod is gathering data; power does not need to be cycled, and the change of pod operation takes place immediately. With Switch 3 in the Up position, the trace buffer is less efficiently used, and the disassembly software may be unable to disassemble portions of the recorded data. In general, Switch 3 should be Down.

In the 8086 F mode, when an instruction byte is transmitted separately from a companion byte in the same prefetched word, BHE* and A0 are modified to refer to the proper byte of the split word.

6. MAXIMUM PROCESSOR MODE WITH JUMP POD MODE (8086 JMP and 8088 JMP)

In the 8086 JMP and 8088 JMP modes, the ML4400 records discontinuities in program flow due to jumps, calls, returns, interrupts, etc. At each discontinuity, two entries are recorded in the trace buffer: the Source and the Destination.

The Source is the address and the first byte of the instruction which caused the discontinuity, or was last executed before a hardware interrupt. The Destination is the address and first byte of the instruction which was executed just after the discontinuity. A special status bit is high for the Source entry and high for the Destination entry. Another status bit is low for those Destination entries which are caused by INTR hardware interrupts.

Switch 4 of the DIP switch allows the user to select between recording only discontinuities due to the INTR hardware interrupt (Down) or all discontinuities (Up).

The Jump mode is especially useful for debugging runaway software or following the decision-making logic in a block of code. In this mode, only the first bytes of the instructions on each side of a discontinuity are available to the disassembler; no other bytes of a given instruction are available. Therefore, only the mnemonic can be displayed by the disassembler, if that much; no operands or data cycles are available for display.

For some instructions, the first byte alone does not specify a mnemonic. For these instructions, a guess (indicated by a question mark) is made at the mnemonic, and only the displayed byte value and address are certain. By using the address value, the user can locate the actual instruction in an assembled code listing.

7. TRIGGERING METHODS WITH PREFETCHING OR BYTE REPACKING

The prefetch mechanism of the 8086 or 8088 causes the microprocessor to fetch some bytes from code space which are subsequently discarded due to a jump, call or other discontinuity in program flow. These fetched but unexecuted instruction bytes are unused prefetches, and they present potential problems to those trying to debug code. Typically, the user wants to trigger logic analyzers on executed instructions, and unexecuted instructions can cause false triggering when they are included in the triggering setup.

The Filtered pod mode strips unused prefetches from the samples sent to the ML4400, but the Filtered mode cannot be used for all applications (e.g., Minimum-mode processors), and the 8086 F mode presents some triggering complications due to instruction byte repacking. This section suggests methods to overcome triggering problems created by unused prefetches in the Transparent pod modes, or by repacking in the 8086 F mode.

a. Unused Prefetch Problems in the Transparent Mode

For Minimum-mode processors, or for Maximum-mode processors being recorded using a Transparent pod mode, unused prefetches will sometimes present a problem while trying to trigger the logic analyzer. A few tricks may be used to trigger the analyzer when the instruction is actually executed.

Method One: (Maximum-mode only) Switch to a Filtered pod mode.

Method Two: If possible, set the trigger word to an instruction downstream from the problem instruction, but in the same execution path. To avoid false triggering, this downstream instruction must be far enough away from the cause of the unused prefetch (a jump, call, return, software interrupt, or other interrupting instruction) so that the the downstream instruction is not also prefetched.

Method Three: Trigger on a noninstruction data access unique to the instruction desired. Unexecuted instructions do not perform any noninstruction data accesses.

Method Four: Trigger on a sequence of instruction fetches and/or data accesses unique to the code path desired.

b. Triggering Problems Due to Instruction Byte Separation within a Word (8086 Processor only)

The processor object code comprises instructions quantized in units of bytes; that is, the smallest instruction is one byte long, and instructions range in size up to six bytes, with all sizes in between included. The 8086 usually prefetches words; therefore, in general, one prefetch bus cycle may contain bytes belonging to two different instructions. Even if the bytes belong to one instruction, the processor can be viewed as "using" the bytes at different times, not simultaneously. If the 8086 prefetches a single byte or if (in the 8086 F mode) the pod sends the bytes of a prefetched word individually, the user may have trouble triggering on a particular instruction byte.

The 8086 will prefetch a single byte only when that byte is a high byte (i.e., has an odd address), and when that byte is the destination of a jump, return or other discontinuity (including return from an interrupt service routine which executed between the low and high bytes of a word). If this odd byte is always accessed singly, the user can simply trigger at the odd address with BHE* set to 0. If, however, this byte may also be prefetched as part of a word at an even address, Method Two or Method Three (below) must be used.

In the 8086 F mode the 16I-086 pod can repack the individual instruction bytes of prefetched words into words. The user can somewhat control the degree to which this repacking occurs via Switch 3 on the pod. Unfortunately, the settings of this switch cannot guarantee that repacking will never occur or always occur. Because the pod alters bits A0 and BHE* according to which byte or bytes are sent to the ML4400 in a given instruction record, and since the user cannot absolutely predict that a given pair of bytes will or will not be repacked, triggering on the address of a particular instruction may be difficult.

The following methods are recommended to overcome the problems created by prefetching and repacking uncertainties:

Method One (for triggering on a low byte): Specify the address in the hexadecimal Address field, and place an "X" in the BHE* status bit.

Method Two (for triggering on a high byte): On the Format setup screen, create a binary Address field duplicating the four least-significant address bits (A3-A0, Probes 35-32), then specify the main portion of the address in the hex field, place "0" in the BHE* Status bit of the trigger word, and place an "X" in A0 of the binary Address field. Note that placing an "X" in A0 will cause a "?" to appear in the least significant nibble of the hexadecimal Address field; the values of A3-A1 can be determined by examining the binary Address field. When creating the new field on the Format setup screen, note that the "Sense" of the field should be "Inverted" (INV).

Method Three (for triggering on a high byte): Using two trigger words (for example, A and B), enter the even address of the word containing the high byte in A, and the odd address of the byte itself in B. Place "0" in BHE* of both words, and trigger on "A or B" or similar sequence.

8. DISPLAY

a. General

With the 16I-086 pod in Transparent mode, the State display shows all instruction fetches and data cycles in the same order in which they occurred on the bus. In Filtered mode, the order of cycles in the trace buffer will not match the actual order of cycles on the processor bus (see Section E, "Filtered Mode"), and any unused fetches (fetches following an instruction causing a program discontinuity) are disqualified by the pod and not stored in the trace buffer.

The Disassembly display groups these program fetches and data cycles into assembly language instructions. Regardless of whether or not the pod is filtering unused fetches, these unused fetches are not displayed on the Disassembly screen. The current pod mode is shown in a reversed field at the top of the Disassembly screen.

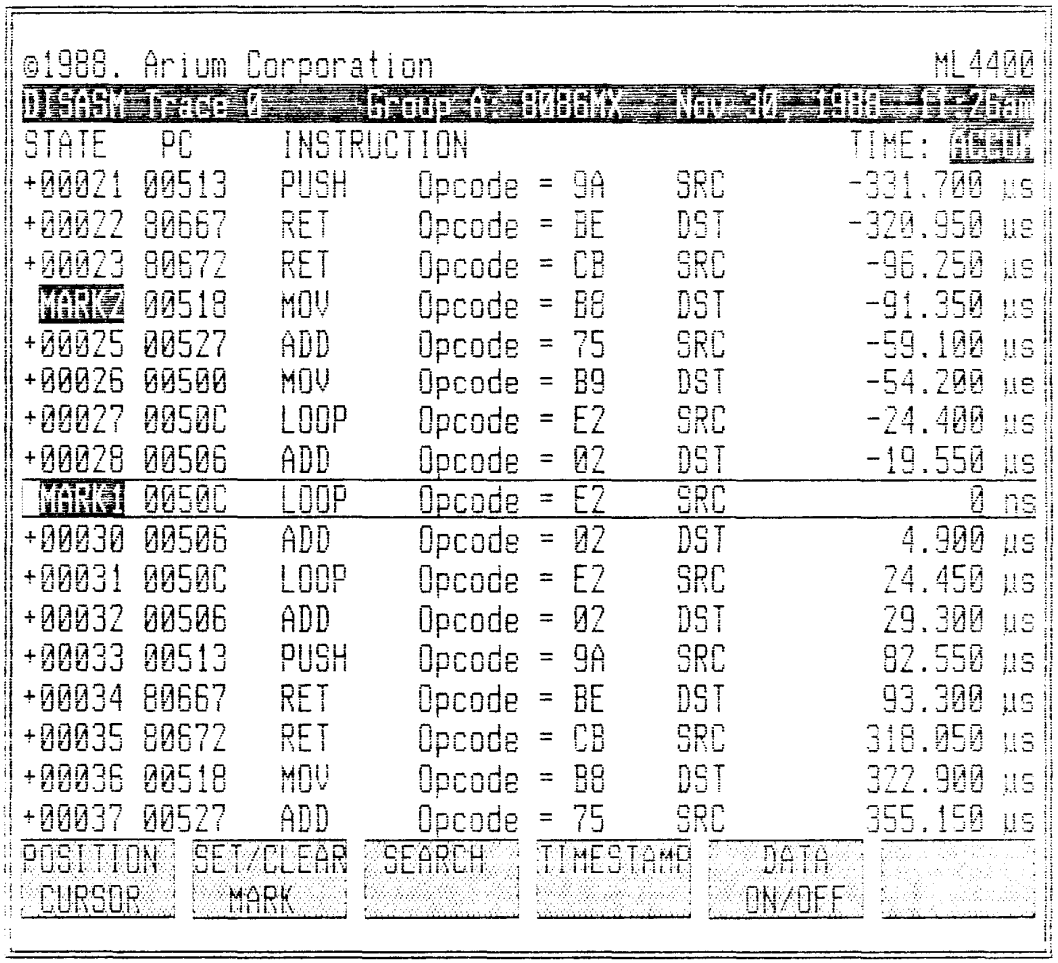
Figure VI-O

DISASSEMBLY DISPLAY SCREEN
FOR 16I-086 POD
(TRANSPARENT AND FILTERED MODES)

```
@1988. Arium Corporation                                        ML4400
DISASM Trace 0 - Group A- 8086MX - Nov 30, 1988 11:23am
STATE   PC      INSTRUCTION                              TIME: DELTA
+00203 00508    ADC     AH,#00                                7.800 µs
+00205 0050B    INC     BX                                    3.450 µs
+00206 0050C    LOOP    00506                                 3.400 µs
+00207 0050E    OUT     #77,AX                                3.400 µs
                 0078<02EC
+00208 00510    MOV     SP,#065F                             11.250 µs
+00211 00513    CALL    80667                                25.400 µs
                 CS:0065E<0000  IP:0065C<0518
+00217 80667    MOV     SI,#0400                             13.700 µs
+00221 8066A    MOV     DI,#F400                              3.400 µs
+00222 8066D    MOV     CX,#0010                              6.850 µs
+00224    REP: MOVSW   [DI],[SI]                            24.400 µs
                 00400>00B8  0F400<00B8
+00230          00402>8E00  0F402<8E00                       11.250 µs
+00232          00404>8BC0  0F404<8BC0                       11.250 µs
+00234          00406>F9D9  0F406<F9D8                       11.250 µs
+00236          00408>8926  0F408<8926                       11.200 µs
POSITION CURSOR | SET/CLEAR MARK | SEARCH | TIMESTAMP | DATA ON/OFF |
```

Figure VI-P

DISASSEMBLY DISPLAY SCREEN
FOR 16I-086 POD
(JUMP MODE)



The formatting and display of data is described generally for the ML4400 in Section IV, "Operation." Further details of data display when using 16-bit microprocessor pods, including typical Disassembly display screens, keyboard functions, and display formats, follow.

b. Disassembly Display Formats

(1) Instruction Display

As with 8-bit microprocessor pod disassembly, each instruction displayed contains a state number corresponding to the instruction position relative to the trigger event, the value of the Program Counter when the instruction was fetched (PC), and the mnemonic and operands for the instruction itself.

(2) Data Cycles Display (including 8087 data cycles)

Data cycles are displayed on lines following the instruction that caused them. Display of these cycles may be turned on or off by toggling the DATA CYCLES softkey.

Each data cycle contains an address, an indication of whether data was read from (>) or written to (<) the address, and the data itself. Each address consists of five hexadecimal characters (20 bits), while data may be a byte (2 characters), or a word (4 characters) for non-8087 instructions. Although the bytes of a word may have been accessed in two separate bus cycles (e.g., an 8086 accessing a word at an odd address or an 8088), the two bytes will be combined into a single word for display.

Some instructions (such as calls, returns, etc.) have some of their data cycles labelled to identify the Instruction Pointer, "(IP)", and CS Register, "(CS)". These labels may appear more than once in the same set of bus cycles, since often the current value is written to the stack and a new value is then read from memory.

The data cycles of repeated string instructions will be displayed, two per line.

The data cycles of 8087 instructions may be as long as 94 bytes. These will be shown on multiple lines, as necessary. The bytes of each separate floating point or integer value will be grouped under a separate address. The 14 bytes comprising the internal status of the 8087, when read or written, will be grouped under a single address. The address will be that of the lowest addressed byte, and that low byte will be the rightmost of the group.

The current version of the 8087 will access only the data segment of memory. If an 8087 instruction is preceded by a segment override prefix, the 8086/8088 will read data from some segment other than DS:; the 8087 then accesses the remaining data bytes in DS:.

(3) Interrupts

Interrupts will be labelled as "EXTERNAL" or "SOFTWARE" (or, in some cases, even more specifically, such as "DIVIDE OVERFLOW", etc.), and their type (vector number) will be displayed. Additionally, the data cycles will be labelled during the stacking of the Instruction Pointer, "(IP)", the CS register, "(CS)" and the Flag register, "(F)". The new Instruction Pointer and CS register retrieved from the vector area will also be labelled.

(4) Question Marks

Occasionally (often just before interrupts) when operating the pod in a Transparent mode, some instructions will be labelled with a question mark. In these instances, the disassembler is unable to determine if these instructions actually were executed.

(5) "UNABLE TO DISASSEMBLE" Messages

Accurate disassembly of the 8086 or 8088 bus cycles requires extensive pattern matching and decision making. For Transparent pod modes, identifying those code segment bytes which are the first bytes of instructions is a difficult task requiring the disassember to make initial guesses concerning the first bytes of instructions and the location of data cycles relating to a given instruction. These guesses are modified until the trace buffer is disassembled without any discrepancies in fitting known patterns to each instruction.

The Filtered modes are simpler to disassemble, due to the presence of "first-byte" status bits on instruction bytes, but locating the noninstruction cycles associated with certain instructions still requires iterative matching of patterns throughout the trace buffer. The only recording modes which do not require extensive effort to disassemble are the Jump modes.

Occasionally, the trace buffer may contain a singularity (an abrupt, small irregularity) through which disassembly cannot pass without a pattern discrepancy. Such a singularity can arise from the use of data qualification, or the resetting of the microprocessor while recording. The disassembler will recover from these singularities in several ways, depending mostly upon how large a piece of the trace buffer was successfully pattern-matched just preceding the singularity, and whether or not there are signs that the processor was reset.

If a reset occurred, the disassembler will display a line with "RESET" in the Address field, and then the pattern matching will resume past the singularity. If the singularity is not due to reset, and a significant number of instructions matched their specific patterns before the singularity, the dissassembler will display "UNABLE TO DISASSEMBLE XXX CYCLES," where XXX is the number of confusing cycles. Disassembly will proceed beyond the singularity.

(blank)