HEWLETT-PACKARD CO.

IOERS

Ltr	Revisions	Date	Appr'd
A	As Issued	12/20/82	DH

			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LT	P.C. #	APPR / DATE	APPD	SHEET # 1 OF 80
	REVISIONS		SUPERSEDES	DWG # A-1FE1-3030-9

TABLE OF CONTENTS

1 INTRODUCTION

1.1 OVERVIEW

The input/output channel for the FOCUS System is the IOP (Input/Output Processor). The IOP is a valid implementation of a subset of a CSG Channel Adapter. The IOP will support all CSG Device Adapters. The mechanisms of I/O include direct I/O by the CPU, CPU interrupts, MPB Command execution by IOP, and direct memory accesses (DMA) by the IOP. A FOCUS system may contain up to six IOP's, each having an addressing capability for eight interfaces.

The IOP is controlled by commands sent over the MPB (Memory-Processor Bus) from the main CPU or a CPU dedicated to I/O and/or other special tasks. Data transfers independent of the CPU are controlled by eight DMA resources in the IOP.

The IOP external I/O bus is a synchronous bus capable of 8 or 16 bit transfers with separate address and bidirectional data lines.

A periodic interface poll is used for detecting interface service and data requests. An affirmative response to the poll will be interpreted as a data request if the poll winning interface has its DMA mask bit set. Checking status will determine if the request was for a DMA transfer or an interrupt since a DMA transfer request will only occur if status is true.

By loading the IOP registers with the appropriate information and starting DMA, data can be transferred independent of the CPU. Data will be transferred by bytes or by half-words until a count has been exhausted, a status error is detected, Device End is asserted, or optionally a character (byte mode only) match is detected. An interrupt request will be recorded at the end of a DMA operation or at an error condition.

MPB Command chaining is possible by setting a bit in the Status register. A list of write or control MPB Commands will be executed by the IOP one per poll cycle if no DMA is requested. The fourth word of the DRT points to the list. If the Status register bit is not set, normal interrupts occur.

Interrupt requests for each interface can be masked individually on the IOP and all CPU interrupt requests from an IOP can be masked on the CPU. There are sixteen interface-address-independent levels of interrupt. Any number of interfaces may be assigned to a priority level.

1.2 LIST OF I/O STATE INFORMATION

The bit numbering convention used in this ERS is consistent with the CPU ERS in that the most significant bit is to the left and it is the lowest numbered. The I/O bus, however, is numbered with the MSB to the left and it is the highest numbered.

Previous interfaces refered to addressable registers as R4 through R7 corresponding to IC addresses 0000 to 0011. All future references will be FR0 thru FR15 corresponding to IC address 0000 to 1111.

A FOCUS system may contain up to six IOP's, each with an interface addressing capability of eight. Every interface has a four word entry in the system's memory-resident Device Reference Table (DRT). The address of the first word of a DRT entry is computed as follows:

00C CCP PP0 000 (byte address)

Where CCC is the Channel Number of the IOP and PPP is the Peripheral Address of the Interface. The first channel is number 1. Each device entry has four words which contain the following information:

Link / Operating System Information
Data Segment Pointer
Interrupt Service Routine Pointer
MPB Command List Pointer

The first word contains link information for interrupts and operating system information. The second word contains a data segment pointer which points to the global data segment used by the interrupt service routine. The third word contains a program pointer which points to the interrupt service routine for the device. The fourth word contains the MPB Command List Pointer.

2 IOP OPERATION

2.1 SYSTEM COMMUNICATION

Communication between IOP's and CPU's or memory is done over the Memory Processor Bus (MPB). (For a detailed description of the MPB and its protocol see the Memory Controller ERS.) A channel number is assigned to each IOP from the range 1 to 7.

The IOP uses byte, half-word, and word transactions with memory on the Y channel and word transactions on the X channel. In addition to reads and writes the IOP will also use semaphore operations with memory. The IOP will receive message bits from memory to indicate double bit memory errors, slave address errors, slave data errors, and slave write data valid.

The IOP communicates with CPU's by receiving channel-to-channel communications (MPB commands) and it can write to a CPU using channel-to-channel communication in response to a request for data. It can respond to an I/O device's need for service by sending the CPU a channel message (an interrupt). The channel message may be a broadcast message if no specific CPU channel address is stored on the IOP.

System Interrupts (Sys. Int. in Fig. 2.1) refer to message bits 0-2 and 6-15. Defined bits that IOP recognizes are: 0 for double bit memory errors, 1 for slave address error, 2 for slave data error. The IOP sends the following messages: 7 for illegal MPB Command, 8 for MPB error, 9 for double bit memory error, 13 for timer interrupt, 14 for CSG attention requests, 16-31 for I/O interrupts.

2.2 INTERFACE POLLING

Interfacing polling is the background activity for the I/O bus (See Figure 2.1). A poll occurs when the IOP sets INT true and latches the IOD lines some time later. An interface requesting service will set an IOD line corresponding to its PA true. Whenever an I/O operation is completed, interface polling is begun to allow interfaces to request service.

Because responding interfaces may be requesting either a DMA data transfer or an IOP MPB command or a CPU interrupt, those

interfaces enabled to do DMA and enabled for interrupt are given a DMA transfer unless the STS line is asserted false. If more than one interface enabled to do DMA or interrupt responds, then the highest PA responding interface will receive service first.

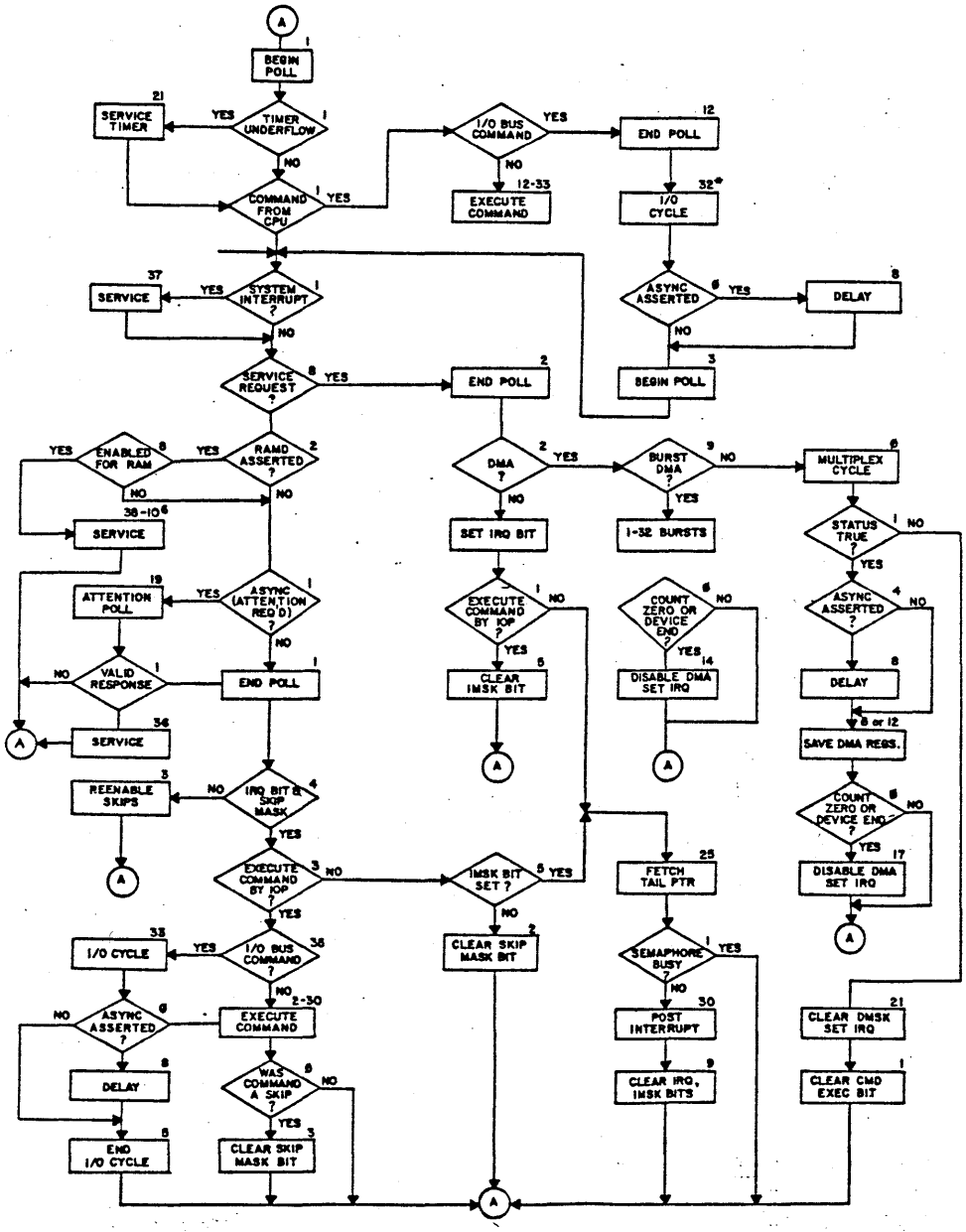


FIG 2.1

			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
P.C. #	APPR	DATE	APPD	SHEET # 8 OF 81
REVISIONS			SUPERSEDES	DWG # A-1FE1-3030-9

2.3 DIRECT I/O

Direct I/O by a CPU is performed when the CPU encounters an I/O instruction and translates that instruction into MPB commands. The MPB command is sent to the IOP's slave address register, with data sent to the slave data register. Contained in the information sent to the slave address register will be IOP destination, the peripheral address of the interface, the operation to be performed, and, if the instruction was a read, the CPU's return address.

Before a transfer takes place, it is necessary for the I/O driver to check status and flag to insure the interface will be able to receive or send data when the bus is strobed. Reading from an interface instructs the IOP to send a data word to the CPU when the I/O read operation is complete.

2.4 DIRECT MEMORY ACCESS

As shown in Figure 2.2, the interface poll response, IOD 0-7, is AND'ed with the DMA Mask register to determine whether a request by an interface enabled to do DMA exists. If so, the highest priority interface is addressed and responds with STATUS. If true a DMA data transfer is done to the interface and if burst DMA (multiple transfers per poll response) is requested, an entire burst is done. If STATUS is false, the DMA transfer to that interface is terminated. Termination of DMA for any reason (other than by CPU) causes a bit in the Interrupt Request register to be set to indicate an interrupt request from the appropriate PA.

If no interface requests service by poll response, the RAMD line is checked for a valid response. If a true response exists and if RAMD is enabled (MSB of DMAPA = 1) then the PA in the DMAPA register will be enabled for a data transfer. A single data transfer or multiple transfers can take place when RAMD is true. All RAMD transfers are done through FR8 or FR10. The intended use of this line is to allow compatibility with Greyhound and it should NOT be used in future interface designs.

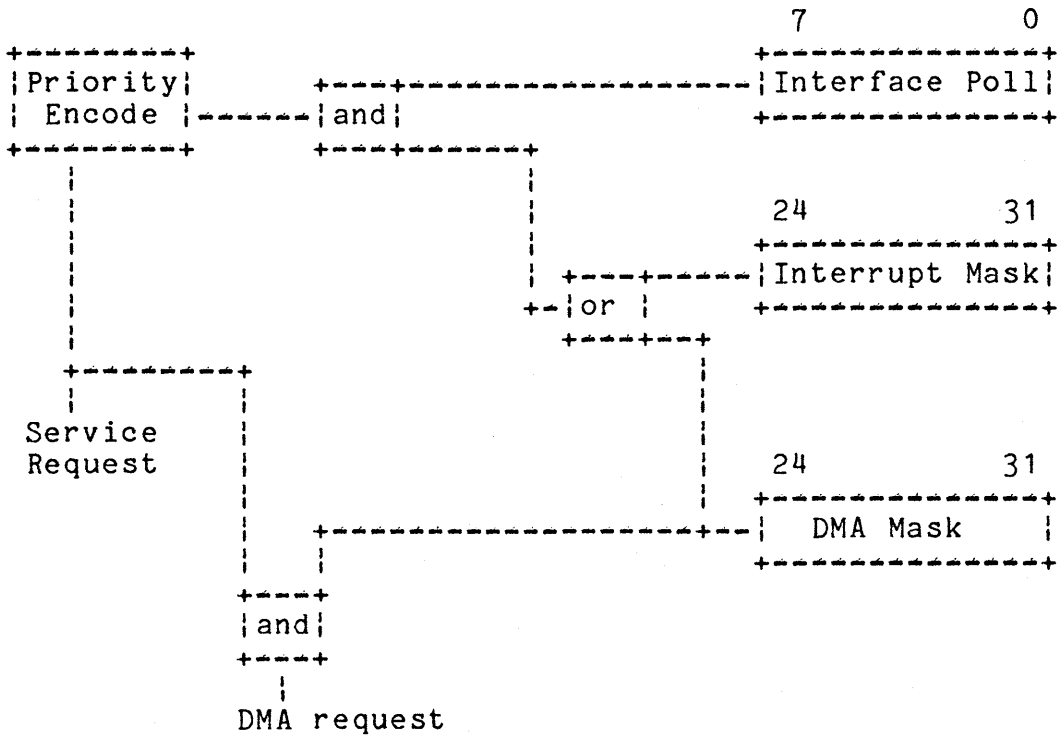


Figure 2.2 Valid Interrupt Request

2.5 INTERRUPT REQUESTS

Valid interrupt requests are either sent to the CPU or processed by MPB command list by the IOP, depending on a bit in each PA's status register. The highest responding PA may request either DMA or interrupt and be given service.

Interrupt requests result from four kinds of events: a response to the interface poll on the I/O bus, termination of a DMA block transfer, setting the interrupt request bit by MPB command, and events not related to one of the eight PA's. These latter events include the interval timer interrupt , memory errors on accesses by the IOP, and attention request interrupts and they are always sent to the CPU as interrupts.

CPU interrupt requests, except from memory errors or from

the timer, can be masked individually on the IOP or all I/O interrupts can be disabled at the CPU. In addition to this, each PA on each channel is assigned an interrupt priority level, 0-15, which is determined by software and stored on the IOP. The CPU automatically masks interrupt requests whose levels are equal to or less than the level of the interrupt service routine that the CPU is executing. This creates sixteen level interrupt structure with additional masking of individual devices and the option of operating with I/O interrupts disabled.

Interrupt requests serviced by the IOP are handled by executing one MPB command per poll when there is no valid DMA request and when a bit in the status register requests MPB command execution by IOP.

2.6 I/O BUS

2.6.1 Functional Description of I/O Bus Lines

IOP provides an I/O bus which has features compatible with 980XX interface cards and with the CSG IO cards. Interrupt lines (IRH, IRL) have been eliminated in favor of doing an interface poll periodically. Immediately after power on, POUT, NDOUT, NRESET, NIOSB, and NINT are initialized active.

The I/O bus has the following lines:

No.	CSG	Old	Description
16	DB[15:0]	DIO15-0	Data Lines
3	PA[2:0]	PA2-0	Peripheral Address
1	BP[1]	IC4	Bus Primitive/ Interface Control
1	CBYT	IC3	Channel Byte
1	CEND	IC2	Channel End
1	BP[0]	IC1	Bus Primitive/ Interface Control
1		FLG	Interface Flag
1		STS	Interface Status
1	IOSB	IOSB	I/O Strobe
1	POLL	INT	Interface Poll
1	DOUT	DOUT	Data Direction (true=out of IOP)
1	BR	BR	DMA Burst Request
1	IFC	RESET	Interface Clear/Reset
1	ARQ	ASYNC(WAIT)	Attention Request/ Asynchronous Handshake
1	DEND	RAMD	Device End/ Request for Accessing Memory Direct

The sixteen bidirectional data lines are used for transferring 8 or 16 bit quantities in either direction. They are also used by interfaces to respond to a poll.

The three peripheral address lines are used to select one of eight interfaces. When an interface recognizes its address, it immediately asserts status and flag.

The four interface control lines are used to inform the interface about the destination or source of data on the next

strobe.

The Flag line exists for compatiblity and is used only during MPB Command to read flag. A true response on the flag line is the interface's assurance that it will allow access to the addressed register on the next strobe.

The Status line is used to signal the IOP that an exceptional status condition exists on the interface (STS = Card Address * Card in Slot * Not Exceptional Status). Status must be true for a DMA data transfer and any status false after a request for service will result in DMA termination and an interrupt request bit set.

The trailing edge of I/O Strobe can be used by an interface to latch data on a data out cycle. During burst input, the trailing edge of strobe indicates the next data may be placed on the data lines.

The Interface Poll line is asserted by IOP between I/O bus cycles. The purpose of this line is to check the interfaces for service requests which may result in an interrupt request or a DMA data cycle. An interface responds by driving the data line corresponding to its PA. An interface must not change its response (either a request or non-request) during a poll cycle. IC4 (BP[1]) determines if the poll is a Service Request poll (IC4=0) or an Attention Request poll. IC1 (BP[0]) is zero during a Service Request poll.

The Data Direction line determines the direction of data transfer and is used by the interface to know when to drive the data lines.

The DMA Burst Request line is used to request a DMA burst during a DMA transfer. Bursts are terminated when Burst Request goes false or when 32 transfers take place, whichever occurs first. The length of I/O Strobe is shortened during a burst transfer. The assertion of Burst Request implies a valid status condition.

The Reset line is used to initialize the interfaces to a known state (Interface Clear). The line can be asserted by MPB Command.

The Async (Wait) line allows an interface to extend the

length of I/O strobe. Async is ignored when Burst Request is true. The extension is eight states. Async is also used during poll cycles as a request for an Attention Poll (See Attention Request).

The Request to Access Memory - Direct line is used to request a DMA transfer without responding to a poll. The line is tested for a valid response at the end of checking for a valid DMA poll response. Only one interface should be enabled to respond on this line and that interface's PA is stored in the DMAPA register. This line is also used as Device End (See CSG I/O Standards) in a time multiplexed way. During an Interface Poll, the line is used to detect RAMD requests. When Interface Poll is false, the line is used to detect Device End (except when RAMD burst transfers are taking place).

The POUT line controls the direction of the external buffers. It also enables the IOD lines from the IOP internally. This line is positive true.

The line DB5 is used to turn the self test LED on and off. It also controls the enable for the input buffers. If DB5 is high, the LED is on and the inputs to IOP are disabled.

2.6.2 Electrical Specifications

The I/O Bus lines have negative true logic. This means a low voltage is an active signal and a high voltage is an inactive signal. The one exception to this is the POUT control line which is positive true logic.

The recommended external buffers are the 8304B Bidirectional Transceivers. They are fast enough to allow a four state burst mode at 18 MHz and provide MOS logic levels (3.6 v. min.) required by the IOP. Figure 2.3 shows the IOP to external buffer interface.

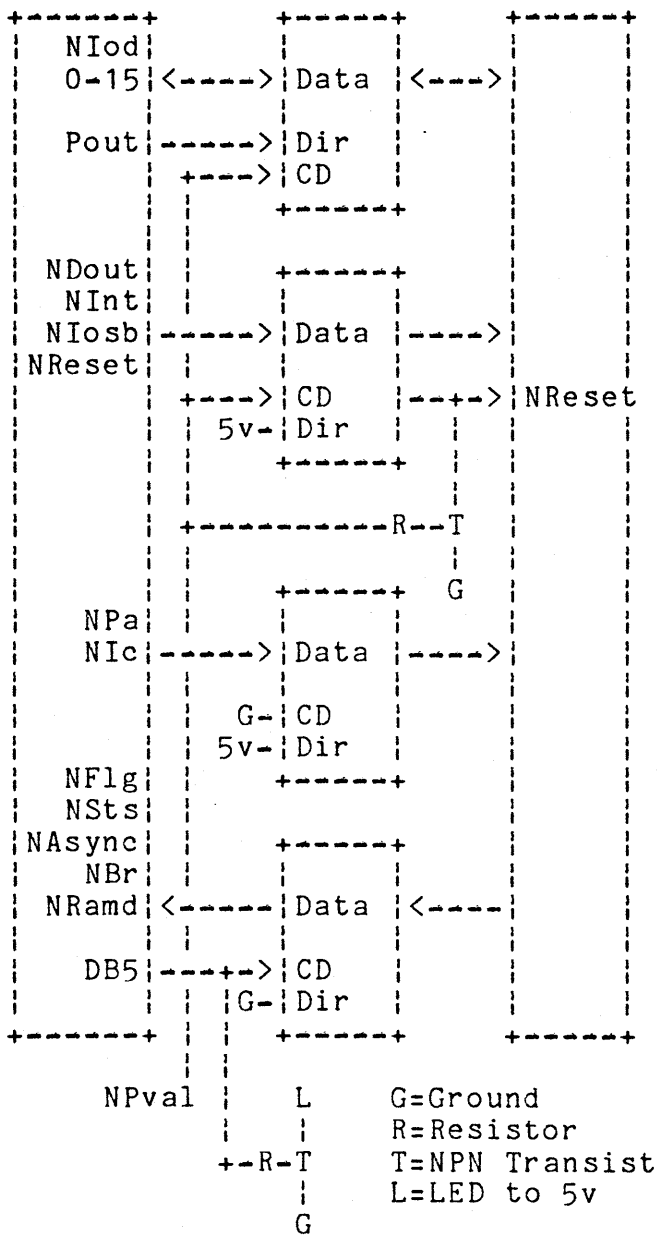


Figure 2.3 I/O Buffers

				MODEL	STK #
				I/O System ERS	
				BY Fred Gross	DATE 09/20/82
LT	P.C. #	APPR	DATE	APPD	SHEET # 15 OF 80
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

3 DIRECT MEMORY ACCESS

3.1 DMA TRANSFERS

There are eight logical DMA register sets, one for each peripheral address, and one active set which contains information for the DMA currently active. The logical sets hold information until an interface currently doing DMA wins a poll. At that point the information is transferred to the active register set for a single transfer or a burst transfer. When the transfer is complete, the logical registers are updated with the new values contained in the active registers. DMA cycles may be single transfers or bursts of up to 32 transfers. During halfword DMA transfers the interface control lines will be false (IC = 0000) except for the last transfer (IC = 0010) if specified. Byte DMA transfers will be through FR4 (IC = 0100) except the last transfer (IC = 0110) if specified. (NOTE: Two bits in the Status register control setting Channel End (IC2), one applies to multiplex DMA and one applies to burst DMA. Their sense is NOT the same.)

Each logical register set consists of a Current Address (CA*) register, a Current Count and Status (CCST*) register, a Termination Field (TF*) register, and a Data Buffer (DB*) register. The CA*, CCST*, and TF* registers are loaded by MPB command.

The starting address should point to the first memory location involved in the transfer and the count should be the exact number of transfers desired. Data packing by the IOP on half-word transfers means that the IOP has one or two half-words in its data buffer once DMA is underway. Byte DMA does not use data packing and as a result it uses four MPB cycles per word transfer versus one for half-word DMA. At the successful completion of DMA, the count register will be negative and the address register will point to one or two locations beyond the last DMA location.

A DMA cycle may be ended in five ways. 1) When the count register reaches zero, DMA is terminated and an interrupt request bit is set. 2) If the match bit in the status register is set and if a match occurs for byte DMA, DMA is terminated and an interrupt request bit is again set. Alternatively, a verify mismatch can end DMA (match and verify are mutually exclusive options). 3) If an interface requests

DMA with STATUS false, the DMA will be terminated. If valid data remains in the data buffer, it will be written to memory. The Command Execution bit will be cleared and the Interrupt Request bit will be set. If interrupt is enabled, an interrupt posting will be attempted immediately. 4) If a device asserts Device End during DMA, DMA will be terminated and an interrupt request bit is set. 5) The CPU may end DMA by command, in which case an interrupt request bit is not set.

NOTE: DMA'ing to within four bytes of non-existent memory may cause the IOP to send a MPB error message bit to the CPU.

For all DMA's except those using RAMD, the following apply. Current count is two less than the number of transfers remaining once a Start DMA has been issued. If DMA terminates on count and it was a multiplex transfer, count will be -2, if a burst, count will be -2. If DMA terminates on Device End, the number not transferred is count+2. If DMA terminates on match, the number not transferred is count+2. If DMA terminates on verify failure and the last transfer was a multiplex, the number not transferred is count+2, if burst, the count is the same as multiplex, but one additional unrecorded transfer took place. If DMA terminates on Status false, the number not transfered is count+2.

For RAMD transfers, the count is one less than the number of transfers remaining once Start RAMD was given and the final count is -1.

3.1.1 DMA Multiplex vs Burst

The sequence of events for a single or multiplex transfer is as follows. A device wins an interface poll and its status is true. Its logical registers are loaded into the active registers, a transfer takes place, the logical registers are updated, and a new poll is conducted.

Burst cycles are similar to single transfers except the interface asserts the Burst Request line to indicate that a burst cycle will occur. Asserting the Burst Request line implies a valid status condition.

A burst of up to 32 data transfers of bytes or half-words will take place when in burst mode. If an interface does not need 32 transfers it can set the BR line false to terminate

DMA at less than 32 transfers. An interface may immediately request another burst, but to do so it must win the next poll. Maximum I/O transfer rate is a burst of 32 followed by a poll, followed by a burst, and so on. If an MPB hang occurs during burst, the cycles will be extended by the number of polls lost times two states.

3.1.2 Channel End

Channel End (IC2) may be asserted when the count register is exhausted. If bit 26 in the Status register is set, Channel End will be asserted at the end of Multiplex DMA's only, otherwise Channel End will not be asserted. If bit 26 is set, Channel End will NOT be asserted at the end of burst DMA only, otherwise Channel End will be asserted.

3.1.3 Device End

All DMA's can be terminated by asserting Device End. Note that asserting Burst Request and Device End is not allowed by CSG IO and doing so will cause unpredictable results for an IOP.

3.1.4 Request to Access Memory - Direct

Request to Access Memory - Direct functions in a manner similar to the DMA line on previous generation processors. Single transfers of half-words or continuous transfers may be requested by asserting RAMD. All RAMD transfers are done through FR8 (IC = 1000) or FR10 (1010). The starting address should point to the first memory location involved in the transfer and the count should be the exact number of transfers desired. A Start RAMD command should be used to initiate transfers. During the last transfer the IC lines will be set to 1010. The intended use of RAMD is to allow compatibility with Greyhound and it is not intended for use in future interfaces.

3.2 DMA OPTIONS

The following options do not apply to an interface using the RAMD line. The DMA input, halfword, match combination cannot be used and Burst Request is ignored on byte verify and byte match.

3.2.1 DMA Verify

Memory data can be verified with data from a peripheral such as a disk by using the DMA verify. The DMA functions as if it were a normal input from an interface, but data is fetched from memory and a half-word or byte compare is made. If a mismatch occurs DMA will be terminated and an interrupt request will be recorded. DMA Verify does not allow terminating the DMA on a match.

The failed memory location is the Start Address plus one or two times the Original Count (OC) minus Final Count (FC) minus three or four.

FL = SA + N * (OC - FC - M) where N = 2 for half-word, 1 for byte, M = 3 for multiplex DMA, 4 for burst.

3.2.2 DMA Match

An input DMA can be terminated when a match occurs between a byte stored in the termination field of a PA and incoming data if the terminate-on-match bit is set. The matched byte will be put in memory. The count register will be checked and a DMA termination-count-zero may occur if the count is exhausted before a match. Termination can occur only on a byte match and it can not be used in burst mode. When a match is encountered, Channel End will be asserted regardless of any bits in the Status register. A simultaneous occurance of match termination and count termination will result in the setting of the DMA-Terminated-on-Match bit only. However, if Device End and a match occur together, both will be indicated in the Status register. The ending address points to the match word and the number of data transferred is SC-EC-2 where SC is the starting count, EC is the ending count.

4 INTERRUPT REQUESTS

4.1 CPU SERVICE

4.1.1 Interrupt Request Register

DMA termination or a non-DMA service request will cause a bit in the Interrupt Request register corresponding to the proper PA to be set. This register is used to initiate either an interrupt posting or command execution by IOP. Bits in the Interrupt Request register may also be set (and cleared) by software to request interrupts on behalf of devices.

If Interrupt request bits are set, the IOP will continuously attempt to service them when no other activity is present on the IOP. The IOP will start with the highest priority IRQ bit and test it for an interrupt or command execution. If the IOP can not do either or if a skip command was executed, a bit in the Skip Mask (in IMSG) will be cleared. The effect is that the IOP will temporarily ignore that IRQ bit until the Skip Mask is restored. Examining IRQ bits and servicing them or clearing skip bits is continued until all bits have been examined in a sequential fashion from highest PA to lowest. The Skip Mask is then reenabled for all PA's and the procedure is repeated. Note that if an Interrupt Mask is written to the IOP, interrupts will be posted from a given (random) PA to lowest PA then highest PA to lower PA until all have been serviced.

4.1.2 Interrupt Mask Register

The interrupt mask is used to prevent interrupt requests from devices not enabled to interrupt or from devices which have interrupt requests pending at the CPU already. A priority encoding by PA of this result provides the highest PA unmasked interrupt request, which the IOP records in memory before signalling the CPU. The interrupt mask bit corresponding to this PA is cleared by the IOP to disable further requests from the same device.

4.1.3 Mapping PA to Interrupt Level

The interrupt level of each device is stored in the CCST register for its PA. CCST[21:4] contains the level number, 0-15. From this number a 16-bit request vector is constructed

with a single bit set to indicate the level. Bit 16 indicates level 15 (highest priority) and bit 31 indicates level 0.

This vector is the right halfword of the address message sent to the CPU to indicate an interrupt request at some level. The left halfword is taken from the Interrupt Message register. This register is initialized by the CPU to contain the left half of an MPB address message which will set the CPU message register. Bits 7-31 must be zero because the Interrupt Message register is OR'd with the request vector to form the address message.

4.1.4 Recording CPU Interrupt Requests

Interrupt requests are recorded in memory by placing the device's DRT entry on a linked list of devices requesting interrupt service. There are sixteen such lists, one per priority level, so the CPU can quickly locate an interrupting device given its level. Figure 4.1 shows the list structure in a situation where three device requests have been recorded.

The list links are stored in dedicated memory locations 2000-2177 (octal) as semaphores as well as absolute address pointers. They are read using the read-and-set-to-minus-one memory operation to insure that only one processor will have access to a list at one time. The list head, if nonzero, points to the first word of the DRT entry for the first device in the list. The left half of this word is a link field which, if nonzero, points to the first word of the DRT entry of the next device in the list. The list continues until a zero link field indicates the last entry. The list tail, if nonzero, points to the first word of the DRT entry for the last device in the list. Additional entries are made at the end of the list.

The sequence of operations that the IOP performs to request an interrupt of the CPU is given below. Calculation of the device number, interrupt request message, and priority level have been described earlier.

1. Read and set to -1 the list tail for the level of the interrupting device until the value read is not -1.

2. Send the interrupt message to the CPU.

3. If the value is not zero, store the address of the first word of the interrupting device's DRT entry in the device's DRT entry link field pointed by the list tail and, lastly, store the address in the list tail pointer.

4. If the value is zero, store in the list head and, lastly, the list tail the address of the first word of the interrupting device's DRT entry.

This sequence completes the IOP processing of an interrupt request. Operation of the IOP resumes as if the request had not occurred, except that the interrupt mask on the IOP now disables the device that requested from requesting again. Note that the interrupt request bit is cleared by IOP during the processing of an interrupt request.

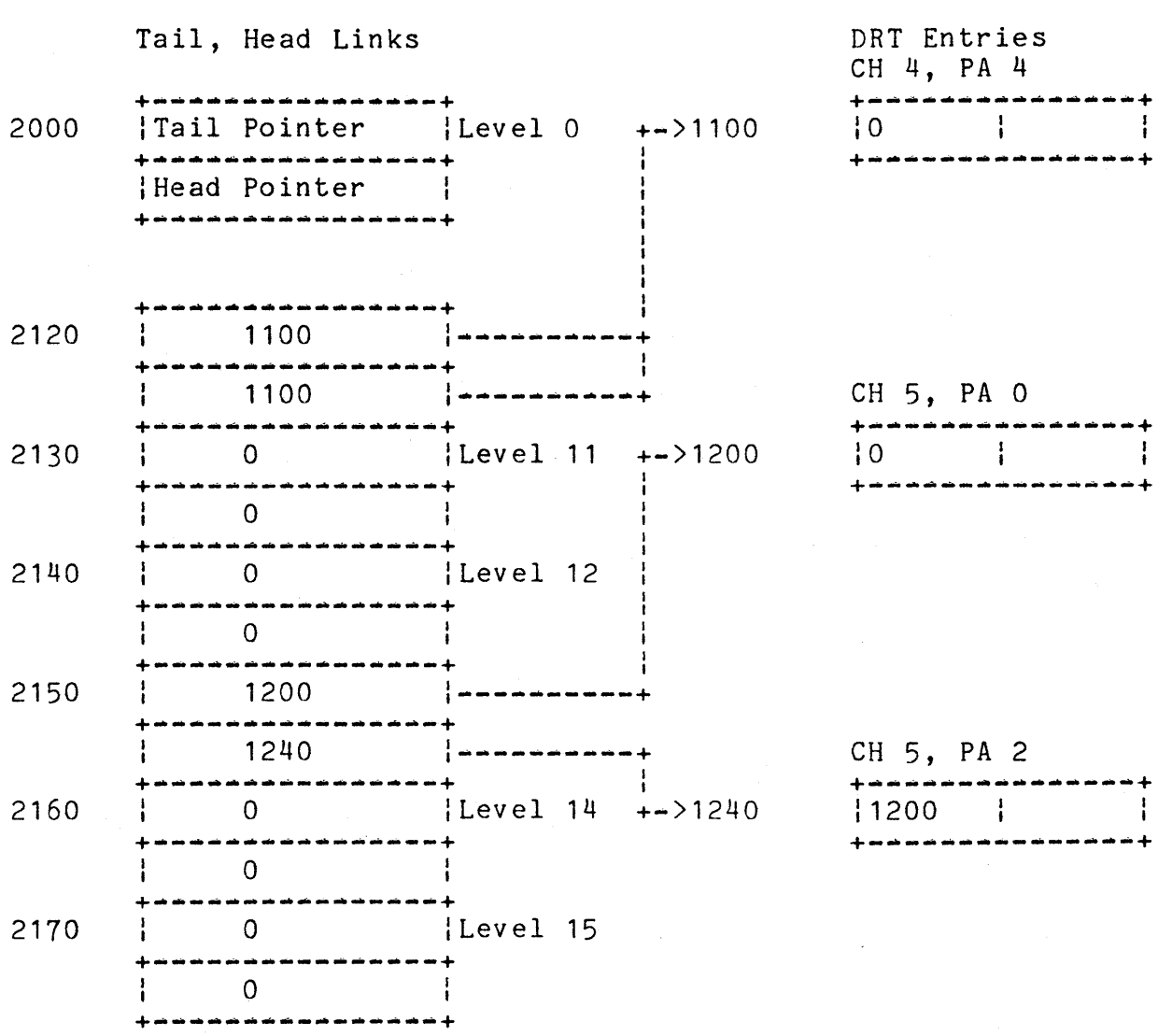


Figure 4.1 Interrupt Linked Lists Example

4.2 CPU OPERATIONS

4.2.1 Interrupt Masking

Although some very long machine instructions can be interrupted, the CPU generally ignores I/O interrupt requests until the end of a machine instruction. At that time, if a message bit corresponding to an I/O interrupt has been set, if the corresponding bit in the CPU Interrupt Mask is clear, if the interrupt enable bit (I) of the CPU Status register is set, and if no higher priority interrupt request or trap exists, an I/O interrupt occurs. Figure 4.2 illustrates this masking process except for the effect of higher priority interrupts and the lockout during machine instructions.

If an unmasked request at level n is the highest priority interrupt request, then an interrupt at level n is granted. The microcode interrupt handler will save the interrupt mask and then update the mask to disable interrupts from priority levels n and lower. Levels higher than n which were previously disabled will remain disabled. This mask will be in place on entry to the service routine for the device at level n. The old mask will be restored on exit from this service routine.

4.2.2 Identifying the Interrupting Device

The sequence of operations that the CPU performs to obtain the device number of the device to be serviced is given below. The interrupt level to be serviced is that of the highest priority message bit that was set and not masked, as described above.

1. Read and set to -1 the list tail for the level to be serviced until the value read is not -1. Read the head pointer. Since this is a pointer to a device's DRT entry, the device number is found in bits 22-27.

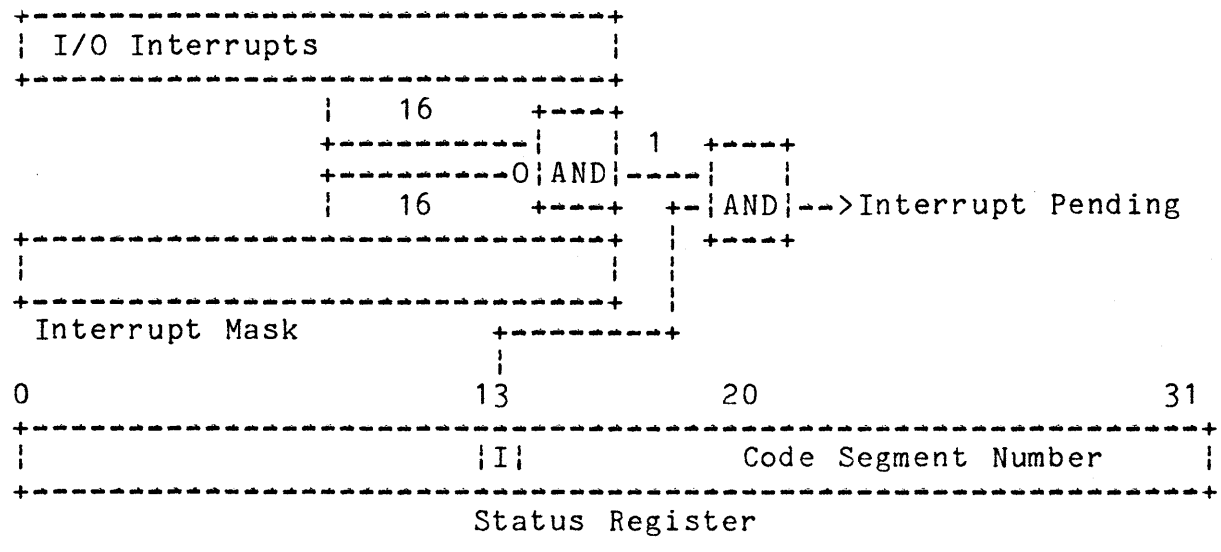


Figure 4.2 CPU Interrupt Masking

2. Read the link field in the word pointed to by the list head.

3. Clear the message bit in the CPU message register for the level to be serviced if the link field was zero, indicating that no more requests are pending at that level.

4. If the link field was zero, store zero in the list tail. If the link was not zero, store zero in the link field, store the link field in the list head, and, lastly, restore the tail pointer.

This process results in the removal of one device from the list of requesting devices and gives the CPU the device number required to access the DRT.

4.2.3 Reaching the Service Routine

The DRT entry contains pointers to the interrupt service routine and variable area (See Figure 4.3). Using this information, the microcode interrupt handler swaps out of the CPU the current program environment and sets up the

environment of the service routine. Refer to the machine instruction set ERS for the details of these CPU operations.

The interrupt service routine is entered with interrupts disabled. The CPU interrupt mask has already been updated to mask interrupt requests of equal or lower priority.

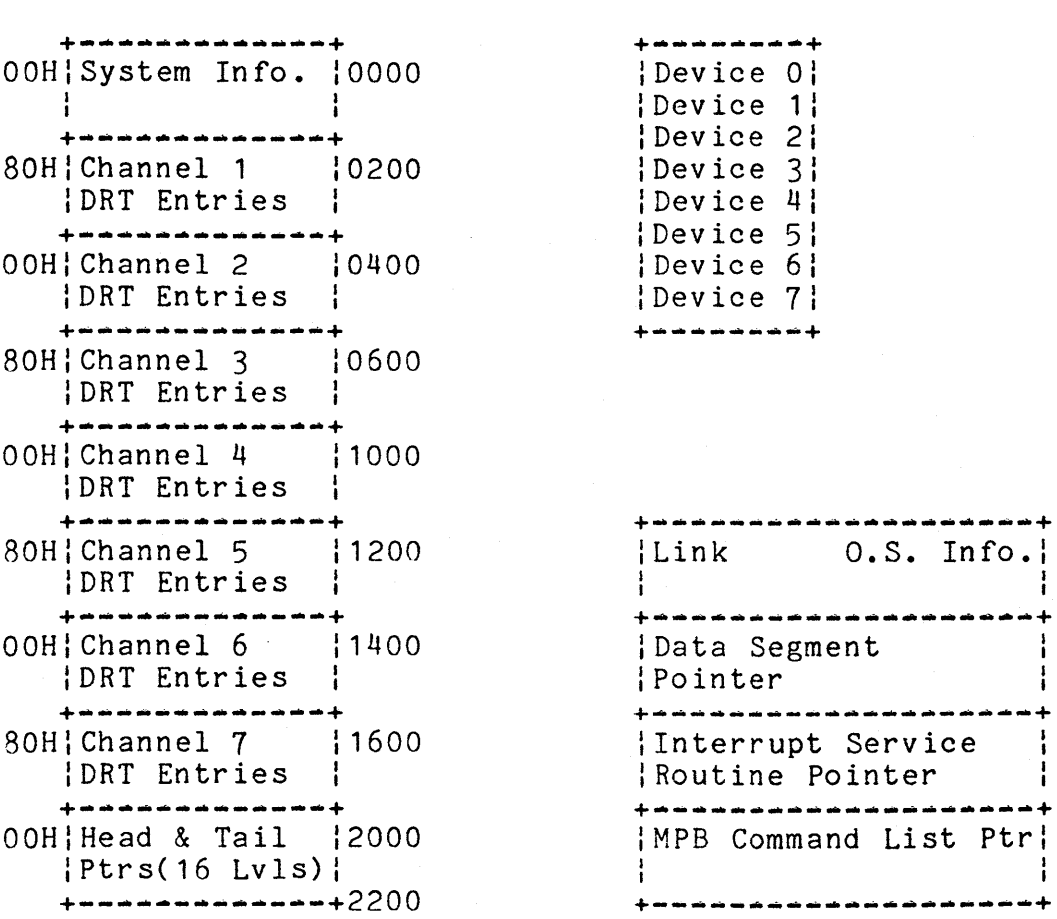


Figure 4.3 DRT Entries

4.2.4 Interrupt Routine Exit

At the completion of the service routine, the CPU microcode for the interrupt exit instruction restores the environment of

the interrupted program or the environment of the dispatcher. Refer to the Machine Instruction Set ERS for the details of these CPU operations.

Interrupt service routine exit has no effect on the I/O channels or the interrupt request structure except that the CPU interrupt mask corresponding to the interrupted progm (or dispatcher) is restored.

4.3 MPB COMMAND EXECUTION BY IOP

MPB commands can be executed by the IOP independent of the CPU. A bit in each status register for each PA specifies whether an interrupt will be sent to the CPU or whether the IOP will execute an MPB command from a list in memory.

The command list location is stored as the fourth word of the DRT. It is updated as the commands are executed. The command lists consist of pairs of command followed by one or two data words. One Write or Control command is executed each poll cycle where no higher PA is requesting service. Read commands are not allowed in the command list, since using them will cause unexpected channel communication. For the Write Interface and Read Interface & Record commands, the PA field is taken from the command word when executing the command. All other commands use the PA from the channel executing the command.

If a skip command is executed, that PA will be disabled from executing commands (or posting interrupts) until a poll cycle of no activity (i.e. no DMA's, no interrupts, no other PA command execution, no RAMD request). The Skip Mask is located in the lower eight bits of the Interrupt Message register.

The interrupt request bit must be set or the interface must respond to a poll to activate Command Execution. A Status false during DMA will cause the Command Execution bit in the status register to be cleared. Commands in a list must have the correct PA in the appropriate field.

The execution times for commands executed by IOP are approximately from one every 76 states to one every 104 states for most instructions.

4.4 ATTENTION REQUESTS

During a Service Request Poll cycle (INT true) assertion of ASYNC will be treated as a request for an Attention Poll. If there is no response to the Service Request Poll and if RAMD is not asserted, then ASYNC will be tested for an Attention Poll Request. If true, IC4 (BP[1]) will be set true with INT true for an Attention Poll.

A poll response will be latched and priority encoded according to PA. The winning response (if any) will then receive a write of 1 to FR1 (IC=0001). This corresponds to an Attention Request Disable. If no device responds to the Poll, IOP will just continue with Service Request Polls.

The PA receiving an ARD will be logged in TF0(16:8) according to PA. A mask (in TF0(0:8)) is used during the Attention Poll to exclude any unwanted PA's. See CSG I/O Standards for further details.

5. MPB I/O COMMANDS

Transfer of commands and data between the CPU and IOP is done with MPB channel-to-channel operations. These transactions consist of sending a channel address and a single data word in primary mode over the MPB for a WRITE operation. The return address for the requested data word is contained in the channel address for READ instructions. The write of data back to the CPU is done in secondary address mode. A third type of I/O command does not require a data word at all. These CONTROL operations are like WRITE's except a zero data word is sent and ignored by the IOP.

The IOP interprets the channel address as an I/O command and uses or returns a data word based on that command. The format of the channel address for an I/O command is given in Figure 5.1. Not all of the fields are used by the IOP for every operation but they exist for uniformity. Only the IOP whose channel number appears in the channel number field receives and executes the command.

NOTE: The bit positions in the commands refer to the MPB channel address and data words. The CPU shifts the IOP command bit fields and forms the MPB I/O Commands in microcode.

Figure 2.1 showed that the MPB commands are executed periodically by the IOP. Until execution of a command is begun, the IOP will not accept another command.

The MPB commands are listed below with their operation codes in octal. In all cases the format of Figure 5.1 is used. After the name of each command is a R, W, or C to indicate whether the operation is a READ, WRITE, or CONTROL. Additional commands for IOP command execution are unlabeled. Figure 5.2 contains a summary of the commands. Undefined op codes will cause a message to be sent (bit 7) to the CPU and otherwise be ignored.

The MPB commands are addresses each pointing to 1 of 128 words of control store on IOP. The first 128 words form a jump table to the actual command routine to allow adjustments in IOP microcode without affecting MPB command code assignments.

5.1 DMA COMMANDS

5. Write DMA Start Address (W) WDA

The data word is loaded into the Address register corresponding to the PA in the channel address. It is interpreted as an absolute byte address and for halfword DMA must lie on a halfword boundary.

6. Write DMA Termination Field (W) WDTF

The data word is loaded into the TF register corresponding to the PA in the channel address. For byte input bits 24-31 specify the termination byte. Any bits in the data word 0-23 will disturb the adjacent termination field or other data in the TF register.

3. Write DMA Status (W) WDS

The data word is loaded into the CCST register bits 16-20, 25-31 corresponding to the PA in the channel address. The interrupt field is assumed zero and the old level is unchanged. Since the DMA status bits control DMA operation, this command should only be used when DMA is disabled. Table 5.1 lists the status bits and their meanings.

4. Write DMA Count (W) WDC

The data word is loaded into the Count register corresponding to the PA in the channel address. Bits 16-31 are interpreted as the 16 bit integer length of the DMA block transfer in transfers.

23. Start DMA (C) SD

A bit in the DMA Mask register corresponding to the PA in the channel address is set to a one. If the DMA is an input, the First Input Pending bit is set. If the DMA is an output or a verify, a data word or byte prefetch is done.

17. Start DMA, Enable Interrupt, Clear IRQ (C) SDEC

Identical to Start DMA except a bit in the Interrupt Mask register corresponding to the PA in the channel address is set to a one and the corresponding Interrupt Request bit is

cleared.

Bit	Meaning
16	DMA Terminated on STS False
17	DMA Terminated on Match or Verify Error
18	DMA Terminated on Count Zero
19	Suppress Channel End on Burst DMA
20	MPB Command Execution by IOP
21-24	Interrupt Level for Device
25	First Input Transfer Pending
26	Assert Channel End on Multiplex DMA
27	F1 F1=0 F2=0 Input F1=0 F2=1 Input, Term. on Match
28	F2 F1=1 F2=0 Output F1=1 F2=1 Input, Verify
29	Byte (=1)/ Halfword Transfers
30	Not used
31	DMA Terminated on Device End

Table 5.1 DMA Status Bits

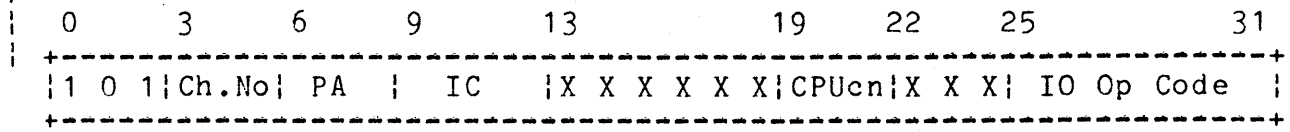


Figure 5.1 MPB Command Format

20. Start DMA, Clear IRQ (C) SDC

Identical to Start DMA except a bit in the Interrupt Request register corresponding to the PA in the channel address is cleared.

15. Enable DMA (C) EDMA

The bit in the DMA mask corresponding to the PA in the channel address is set to enable DMA for the PA.

16. Disable DMA (C) DDMA

The operations are the same as for ENABLE DMA except the bit is cleared to disable DMA for the PA. This termination of DMA does not cause an interrupt request bit to be set.

. Read DMA Current Address (R) RDA

The contents of the Current Address register corresponding to the PA in the channel address are returned to the CPU. It should be interpreted as the byte address of the datum following the datum to be output next for output and as the byte address of the datum last input (or the starting address if no inputs have occurred) for input.

10. Read DMA Current Count/Status (R) RDCS

The contents of the CCST register corresponding to the PA in the channel address are written back to the CPU. Bits 0-15 should be intepreted as the 16-bit integer number of one less than the number of transfers remaining in the current DMA block transfer.Note that the data buffer will contain data after the first or second input transfer. Bits 16-31 are the DMA status bits shown in Table 5.1.

13. Read DMA Mask (R) RDMK

The contents of the DMA Mask register are returned to the CPU in bits 24-31 of a data word. A bit set indicates DMA enabled for the corresponding PA. Bits 0-23 of the returned data word are zero.

14. Clear DMA Mask (C) CDMK
The DMA mask is cleared to disable all DMA. DMA status bits are not affected.

5.2 INTERRUPT COMMANDS

25. Write Interrupt Mask (W) WIMK

The data word bits 24-31 are loaded into the Interrupt Mask register (24:8). Bits 0 to 23 of the data word are ignored.

33. Write Interrupt Message (W) WIMG

The data word, which must be an MPB address message with bits 6-31 zero, is loaded into the Interrupt Message register and it should containt the channel number of the CPU which will service interrupts in bits 3-5.

27. Request Interrupt (C) RINT

The bit corresponding to the PA in the channel address is set in the Interrupt Request register.

30. Clear Interrupt Request (C) CIRQ

The bit corresponding to the PA in the channel address is cleared in the Interrupt Request register.

36. Enable Interrupt (C) EINT

The bit in the Interrupt Mask corresponding to the PA in the channel address is set.

37. Disable Interrupt (C) DINT

The bit in the Interrupt Mask corresponding to the PA in the channel address is cleared.

26. Read Interrupt Mask (R) RIMK

The Interrupt Mask is returned to the CPU as bits 24-31 of the data word. Bits 0-23 are zero.

34. Read Interrupt Request (R) RIRQ

The Interrupt Request register is returned to the CPU as bits 24-31 of the data word. Bits 0-23 are zero.

35. Set Interrupt Level (W) SIL

The data word bits 28-31 (other bits must be zero) are loaded into CCST[21:4] corresponding to the PA in the channel address as the interrupt level for that PA.

5.3 INTERVAL TIMER COMMANDS

40. Write Timer Start (W) WTMS

All 32 bits of the data word are loaded into the Timer Start register to define the time interval at which timer interrupts will be generated. (Note: TC[24:8] will remain unchanged at the update time.)

43. Write Timer Count (W) WTMC

The data word is loaded into the Timer Count register to define the time remaining before the next interval timer interrupt.

44. Read Timer Count (R) RTMC

The contents of the Timer Count register are returned to the CPU. Since this register decrements continuously, the value returned will not be current even when it appears on the MPB.

45. Enable Timer Interrupt (C) ETMI

Timer Disable is cleared to allow timer interrupts.

46. Disable Timer Interrupt (C) DTMI

Timer Disable is set to prevent timer interrupts.

5.4 DIRECT I/O COMMANDS

The IC field asserted on the I/O bus is the logical 'or' of bits 9-12 and bits 25-28 of the channel address word for the Read Interface and the Write Interface commands.

1, 11, 21, 31, 41, 51, 61, 71, 101, 111, 121, 131, 141, 151, 161, 171. Read Interface (R) RIF n

An I/O bus cycle is performed using the PA and IC fields in the channel address with DOUT false to indicate an input operation. The data, IOD 15-0, is returned to the CPU in bits 16-31 of the data word, IOD 15 in bit 16. Bits 0-15 of the data word are zero. STATUS and FLAG are ignored.

2, 12, 22, 32, 42, 52, 62, 72, 102, 112, 122, 132, 142, 152, 162, 172. Write Interface (W) WIF n

An I/O bus cycle is performed using the PA and IC fields in the channel address with DOUT true to indicate an output operation. Bits 16-31 of the data word are output on the IOD 15-0 lines with bit 16 on IOD 15. STATUS and FLAG are ignored.

47. Interface Poll (R) IFPL

An interface poll is done on the I/O bus and the IOD response is returned to the CPU as in the READ INTERFACE instruction.

77. Read Interface Status & Flag (R) RISF

The PA and IC fields of the channel address are used to address an interface with DOUT true. The response on the STATUS and FLAG lines is returned in bits 30 and 31 respectively of a data word written back to the CPU. Bits 0-29 are zero. IOSB is not issued on this bus cycle.

103. Read Interface Status (R) RIST

The PA and IC fields of the channel address are used to address an interface with DOUT true. The response on the STATUS line is returned in bit 31 of the data word. IOSB is not issued.

104. Read Interface Flag (R) RIFG

The PA and IC fields of the channel address are used to address an interface with DOUT true. The response on the FLAG line is returned in bit 31 of the data word. IOSB is not issued.

50. Reset I/O Bus (C) RIOB

The RESET line on the I/O bus is asserted to initialize all interfaces. WARNING: This instruction disables IOP for 10^6 states.

5.5 OTHER COMMANDS

54. Write DMAPA (W) WDPA

The PA in the data word, bits 5-8, are loaded into the DMAPA register. All other bits are ignored. Bit 5 is used as an enable bit, 1=enable 0=disable.

74. Disable IOP Command Execution and Enable Interrupt (C) DCEI

A bit in the status register, bit 20, corresponding to the PA in the channel address is cleared and a bit in the interrupt mask register is set. Command execution by the IOP for that PA is suspended and an interrupt will be sent to the CPU when

the IRQ bit is serviced or when the interface wins a poll.

53. Reset IOP (C) RIOP

The IOP is forced into an initial state from which it begins normal operation by starting an interface poll. The initial state has interrupt request disabled, the Timer disabled, the DMA Mask cleared (all DMA disabled), the Interrupt Request register cleared, and the Interrupt Mask cleared (all interrupts disabled). All other registers remain unchanged and IOP operation resumes immediately.

60. Write MPB Channel Number (W) WMCN

The channel number in bits 3-5 of the data word is written into the IOP's MPB Identification Register. Other bits of the data word must be zero. A Master Channel reset is done when assigning the new channel number.

63. Clear Address Lockout Mode (C) CALM

This command enables the IOP to take more than its allocated share of MPB bandwidth. It can lock out any chip of lower priority.

64. Set Address Lockout Mode (C) SALM

This command prevents the IOP from taking more than its share of MPB bandwidth. The IOP powers on in this mode. It will also increment and replace the count if the count is still negative.

73. Disable IOP Command Execution (C) DCE

This command clears a bit in the Interrupt Request register and clears the Command Execution bit (bit 20 in the Status register) both corresponding to the PA in the channel address. This command is used when a CPU interrupt is not desired at the end of Command Execution by IOP.

70. Initiate Command Execution from Interrupt (C) ICEI

This command sets a bit in the Interrupt Mask register, clears a bit int the Interrupt Request register, and enables Command Execution (bit 20 in the Status register) all corresponding to the PA in the channel address. This command suspends

execution of commands until the interface responds to a poll.

56. Read Data Buffer (R) RDB

This command reads the data buffer register corresponding to the PA in the channel address.

55. Read DMAPA (R) RDPA

This command reads the DMAPA register.

75. Turn LED On (C) LON

This command sets pad DB5 to a high state which turns on the self test LED.
Caution: This command disables I/O bus inputs. Normal I/O operations will not work when the self test LED is on.

76. Turn LED Off (C) LOFF

This command sets pad DB5 to a low level which turns off the self test LED and enables the external buffers.

57. Read Revision (R) RIRV

The date the IOP microcode was released for the IOP currently in the system is returned to the CPU. The format of the data stored in memory is YYYYMMDD in hex where YYYY is year, MM is month, and DD is day.

24. Start RAMD, Clear IRQ (C) SRC

The RAMD enable bit (DMAPA(5:1)) is enabled and the bit in the interrupt request register corresponding to the PA in the DMAPA register is cleared. If the DMA is an input, the First Input Pending bit is set. If the DMA is and output, a data word prefetch is done.

105. Read DMA Termination Field (R) RDTF

This command reads the termination field register corresponding to the PA in the channel address.

106. Write Attention Poll Mask (W) WAMK

This commands writes the data word (24:8) in TF0 (0:8). The remaining bits of the data word must be zero. TF0 (0:8) represents a mask that is used to allow any or all interfaces to participate in a CSG Attention Poll (See CSG I/O Standards). A one in a bit position allows the corresponding interface to participate in the poll. At power on, the mask is set to all ones (all interfaces allowed to participate in poll).

107. Clear Attention Acknowledge Bit (C) CAAK

This command clears a bit in TF0 (16:8) corresponding to the PA in the channel address. If the IOP has issued an AAK to an interface a bit will be set in TF0 (16:8). Message bit 14 will also be sent to the CPU. At power on, all bits are cleared.

110. Read IOP Registers and Suspend (R) RIRS

This command reads and clears the DMA Mask register, the Interrupt Request register, and the Interrupt Mask register. This has the effect of suspending the IOP such that all remaining registers could be read and saved knowing that the IOP will not change state while reading the registers. This command would be useful in a power-going-down situation where resuming operation is desired. The data word returned contains DMSK (8:8), IRQ (16:8), and IMSK (24:8).

113. Write IOP Registers and Resume (W) WIRR

This command loads the DMA Mask register, the Interrupt Request register, and the Interrupt Mask register and is the compliment command to Suspend and Save IOP command.

114. Write RIF Result to Memory (W) WRIF

Used during Command Execution by IOP, this command will do a Read Interface using the PA and IC lines from the command word read from the memory list and record the results in the absolute word address found in the first memory location following the command. If used as an IOW, the PA and IC data is taken from the channel address and the results are recorded in the absolute word address sent as data.

115. Write Count and Status to Memory (W) WCSM

Used during Command Execution by IOP, this command will record the contents of the Current Count and Status register in the absolute word address found in the first memory location following the command. If used as an IOW, the PA is taken from the channel address and the data is stored in the absolute word address sent as data.

116. Read Interface Device End & Burst Request (R) RDEB

This command tests the state of Device End (RAMD) and Burst Request and returns the results. If Device End is asserted, bit 30 is a one, and if Burst Request is asserted, bit 31 is a one.

117. Write DMA Data Buffer (W) WDB

The data word is loaded into the Data Buffer corresponding to the PA in the channel address.

120. Write DMA Current Address (W) WDCA

The data word is loaded into the Current Address register corresponding to the PA in the channel Address.

123. Read Interrupt Message (R) RIMG

The contents of the Interrupt Message register are returned to the CPU.

5.6 INSTRUCTIONS FOR COMMAND EXECUTION ONLY

65. Increment and Branch

This instruction will test the count, the contents of the second memory location following this command, and branch to the absolute address in the first memory location following this command if the value was not zero. It will also increment and replace the count if the count is still negative. If the count is zero, execution will continue at the third memory location following this command.

66. Skip on Status False

This command will test the I/O Status line of the designated

PA and if true, no action is taken. The MPB command list pointer in the DRT points to the next instruction. If Status is false, the skip mask bit corresponding to the PA is cleared. Another command execution for the PA is not allowed until there is no activity on the bus and no interrupts are pending. The command list pointer remains pointing to the skip command.

67. Skip on Flag False

Same as Skip on Status False except applies to the I/O Flag line.

IOP MPB COMMANDS

NAME	MNEM	HEX	OCT
READ COMMANDS			
Read DMA Current Address	RDA	7	7
Read DMA Current Count & Status	RDCS	8	10
Read DMA Mask	RDMK	B	13
Read Interrupt Mask	RIMK	16	26
Read Interrupt Request	RIRQ	1C	34
Read Timer Count	RTMC	24	44
Interface Poll	IFPL	27	47
Read DMAPA	RDPA	2D	55
Read Data Buffer	RDB	2E	56
Read IOP Revision	RIRV	2F	57
Read Interface Status & Flag	RISF	3F	77
Read Interface Status	RIST	43	103
Read Interface Flag	RIFG	44	104
Read DMA Termination Field	RDTF	45	105
Read IOP Registers and Suspend	RIRS	48	110
Read Interface Device End & Burst Request	RDEB	4E	116
Read Interrupt Message	RIMG	53	123
Read Interface FRn	RIF n	(8*n+1)	
WRITE COMMANDS			
Write DMA Status	WDS	3	3
Write DMA Count	WDC	4	4
Write DMA Start Address	WDA	5	5
Write DMA Termination Field	WDTF	6	6
Write Interrupt Mask	WIMK	15	25
Write Interrupt Message	WIMG	1B	33
Set Interrupt Level	SIL	1D	35
Write Timer Start	WTMS	20	40
Write Timer Count	WTMC	23	43
Write DMAPA	WDPA	2C	54
Write MPB Channel Number	WMCN	30	60
Write Attention Poll Mask	WAMK	46	106
Write IOP Registers and Resume	WIRR	4B	113
Write DMA Data Buffer	WDB	4F	117
Write DMA Current Address	WDCA	50	120
Write Interface FRn	WIF n	(8*n+2)	
CONTROL COMMANDS			

				MODEL	STK #
				I/O System ERS	
				BY Fred Gross	DATE 09/20/82
LT	P.C. #	APPR	DATE	APPD	SHEET # 41 OF 80
REVISIONS				SUPERSEDES	DWG # A-1FE1-3030-9

Clear DMA Mask	CDMK	C 14
Enable DMA	EDMA	D 15
Disable DMA	DDMA	E 16
Start DMA, Enable Interrupt, Clr IRQ	SDEC	F 17
Start DMA, Clear IRQ	SDC	10 20
Start DMA	SD	13 23
Start RAMD, Clear IRQ	SRC	14 24
Request Interrupt	RINT	17 27
Clear Interrupt Request	CIRQ	18 30
Enable Interrupt	EINT	1E 36
Disable Interrupt	DINT	1F 37
Enable Timer Interrupt	ETMI	25 45
Disable Timer Interrupt	DTMI	26 46
Reset I/O Bus	RIOB	28 50
Reset IOP	RIOP	2B 53
Clear Address Lockout Mode	CALM	33 63
Set Address Lockout Mode	SALM	34 64
Initiate Cmd Exec from Interrupt	ICEI	38 70
Disable IOP Command Execution	DCE	3B 73
Disable Cmd Exec and Enable Interrupt	DCEI	3C 74
Turn LED On	TON	3D 75
Turn LED Off	TOFF	3E 76
Clear Attention Acknowledge Bit	CAAK	47 107

IOP COMMAND EXECUTION

Increment and Branch		35 65
Skip on Status False		36 66
Skip on Flag False		37 67
Write RIF Result to Memory	WRIF	4C 114
Write Count & Status to Memory	WCSM	4D 115

6. CPU INSTRUCTIONS FOR I/O

All communication between the CPU and an IOP is via the MPB. The CPU instruction set provides the capability to perform these MPB operations as specific instances of generalized MPB operations. This section describes another group of instructions used specifically for efficient communication with the IOP. The remainder of the CPU instructions are fully described in the Machine Instruction set ERS.

Three 16-bit instructions whose mnemonics are IOW, IOR, and IOC are provided for communication with the IOP. They contain sub op-code fields to specify one of many MPB transactions with the IOP. Although the CPU instructions which allow general MPB operations may be used with the IOP, the I/O instructions shorten I/O routines by combining several CPU operations in a single machine instruction.

As explained in Section 5, MPB transactions with the IOP consist of sending a channel address and sending a data word to the IOP. The I/O instructions do these operations using parameters on the stack. The channel address is constructed by combining the I/O operation code, a field of the I/O instruction, with the IOP channel address and interface address from a word in the stack. This provides the efficiency of coding the I/O operation in the CPU instruction with the flexibility to specify the interface address in a data word which is a parameter to the I/O routine.

6.1 I/O INSTRUCTION FORMATS

The format of the I/O instructions are given in Figure 6.1. Bits 8-15 are loaded into bits 24-31 of the channel address as the I/O operation code. It will be possible, with the operation codes, to issue any of the MPB commands of Section 5 using the I/O machine instruction.

6.2 I/O INSTRUCTION PARAMETERS

The I/O instructions use the top word of the stack as the data word for a WRITE operation. The second word on the stack (or the top word for a READ or CONTROL) must be the interface address. Bits 26-28 of this word become the channel number field of the channel address, bits 29-31 the PA field, and bits 22-25 the IC field.

The parameter is popped from the stack on a WRITE and sent as the data word. On a READ operation, the data word returned by the IOP is pushed onto the stack (the address word sent by the CPU contains the channel address used by IOP in returning data). Zero is sent as the data word for a CONTROL operation and it is ignored by the IOP. The interface address is popped from the stack for all I/O instructions. The CPU condition codes are unchanged by I/O instructions.

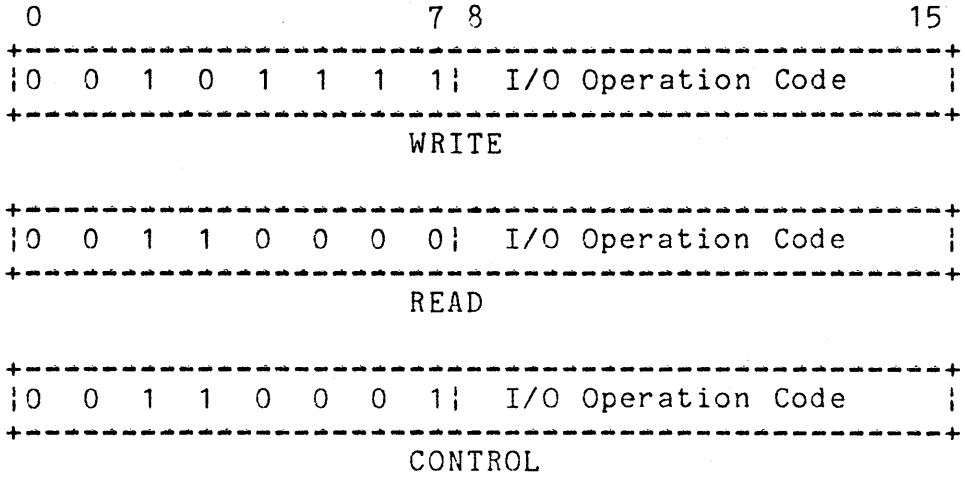


Figure 6.1 I/O Instruction Formats

An IOR instruction will wait up to 6000000 states for data from an IOP. If no data is received in that period of time, the CPU will get a System Error (STT#14) type 3. If an instruction (IOR, IOW, or IOC) has been sent to the IOP, and the IOP has not removed it from the Slave Channel, the following commands will be refused by the IOP. The CPU will then re-try sending the command about 30 times with about 50 states delay between each re-try before causing a System Error.

7. INTERVAL TIMER

The interval timer function is implemented in the IOP with two registers, the second of which decrements and can be loaded from the first. The first register, Timer Start, contains an integer that specifies the length of the time interval. The second register, Timer Count, decrements every two FOCUS system clock states until it goes negative. When it goes negative, an interrupt is requested and the Timer Count register is loaded with the contents of Timer Start to begin counting the next interval.

7.1 TIMER UNDERFLOW

When Timer Count goes negative, the Timer sign bit becomes true indicating an underflow condition. This condition is tested by the IOP microcode periodically if Timer Disable is false (see Figure 2.1). The Timer Disable is stored in a memory cell on the IOP and may be used to disable the timer. If an underflow has occured, bits 0-23 of Timer Start are loaded into the corresponding bits of Timer Count. Bits 24-31 of Timer Start are ignored.

Caution: A negative value in Timer Start can cause IOP to ignore all commands from the CPU.

A second action taken by the microcode on timer underflow is the setting of the Timer Interrupt Request message bit 13 and sending a message to the CPU or sending a broadcast message.

7.2 TIMER ACCURACY AND PRECISION

Because the timer uses the system clock as a frequency reference, high accuracy is not guaranteed. The intended uses of the interval timer are in CPU allocation by the operating system and timing of system execution of programs for accounting or performance measurement.

The interval at which the timer interrupts can be set is from 1 to 2^24 times 28.4 microseconds, set by loading the Timer Start register from the CPU. The length of the interval is exactly (Timer Start(0:24)+1)*512*(one clock period). A loss of one or two states in the timing occurs when the Timer Count register is loaded from the Timer Start. If the interval is set very short and two underflows occur before the

first is serviced by the CPU, the second underflow cannot be detected.

8. INITIALIZATION

The product hardware supplies two lines to the FOCUS system to force the hardware to a power-on (or user self-test) or reset state. They are System POP (SYP) and POP In (PI). Each chip will also generate a POP Out (PO) signal to be connected to the next chip in a daisy-chain fashion. At the completion of a power-on or reset, the IOP starts executing according to its flowchart (see Fig. 2.1).

8.1 POWER ON

When SYP goes to a true state, the I/O microcode is forced to a special trap routine. When PI goes true, PO is immediately set true by the hardware. When SYP goes false, the IOP will run self-test and set its MPB channel number to 0. At the beginning of self test, the IOP will prepare an identification word indicating an IOP chip and that it is bad for the Master CPU and place it in its slave data register. If the IOP passes self test, the bad indication will be changed to good.

When PI goes false, the IOP will set its address mode from secondary (SYP forces address mode to secondary) to primary. Sometime later the master CPU does a read from slave channel 0. The IOP with its primary address mode enabled will send its status word to the master CPU. The master CPU will then write to the IOP's slave a word which contains the channel number it is to be assigned. The IOP sets this channel number into its channel number register, exercises its MPB interface by doing channel-to-channel communication with itself, and propagates the PI signal by pulling its PO pad low.

When the master CPU is done initializing memory it will write to each IOP a message that it should or should not now execute a channel-to-memory test (memory may not be available). After all chips complete their channel-to-memory test, the master CPU will write to all channels an indication that power on is complete and it may begin executing.

8.2 RESET

If PI goes true and SYP stays false for the entire time, a

system reset has occurred. In this case no self-test is done and channel numbers are not reassigned. A master channel reset is done and a set DOUT and clear IOSB, POUT, INT are done.

8.3 SELF TEST GENERAL DESCRIPTION

Self test microcode is made up of 14 independent modules linked with turn on microcode which begins at the power-on trap location. At power on the following sequence occurs:

1) trap to power on routine

2) Execute self test modules 1 through 12. These modules consist of internal IOP hardware tests only. Linkage is through long jump instructions as follows:

Turn on jumps to module 2

Module 2 jumps to module 3

.

.

.

Module 11 jumps to module 12

Module 12 jumps to turn on

Module 1 is a subroutine called by modules 2 through 12

3) Wait for master CPU to assign channel number.

4) Execute self test module 13, then return to turn on.

5) Wait for master CPU to signal start memory access, or skip memory access test. If skip, go to 7.

6) Execute self test module 14.

7) Initialize and begin main I/O microcode execution.

The self test microcode attempts to exercise and test all microcode accessible hardware. Since the hardware tests itself, the code modules are not truly independent. For example, a failure in the bit-set-clear hardware (BSC) will cause the BSC self test code to detect a fault. However, other modules will also detect a fault, if they use the BSC circuitry to test other hardware. For this reason the self test code executes the following instruction when any hardware fault is detected:

```
UNC, JMPL, MCD, HALT, * ; ERROR TRAP
```

This insures the IOP will halt, unless there is a failure in both the 'HALT' instruction and long jump sequencing. Zero is placed in the slave data register (MCD) to insure system turn on halts by signaling the master CPU.

Debug pad 5 is used as a self test indicator signal, and to disable certain external buffers (see Fig. 2.3). During turn on this pad is driven to a high voltage, then driven low after successful completion of turn on. Debug pad 5 is also driven low for a period of 11 clock cycles during execution of self test modules 1, 13 and 14. A total of 13 of these 'progress pulses' will therefore be output during turn on.

Please note that the output control lines on the I/O bus are asserted during certain phases of self test. Care must be taken that this will not adversely affect interface cards during turn on.

In addition to specific hardware tests, each self test module attempts to insure all of its code is executed. In modules 1 through 10 and 12 this is done by initializing the internal timer at the beginning of each module, then comparing it to a ROM constant before exiting to the next module. Modules 11, 13 and 14 maintain a similar count in the MA register under microcode control.

8.4 SELF TEST OBJECTIVES

The following is a list of the types of hardware tests performed by the self test microcode. The objective of self test is to test as much of the applicable hardware as possible, for each test. There are 3 possible reasons applicable hardware would not be tested: 1) the test cannot

be performed with a microcode routine, 2) the hardware is never used by main I/O microcode, or 3) the ROM or development cost of the test would be too high. Since the amount of hardware not tested is minimal, these exceptions are listed after each test.

1) All data cells tested for stuck at "0" or stuck at "1" faults.

2) All data cells tested for R/W ability from all data paths connected to the cell. R/W ability is defined as a successful 0 } 1 } 0 or 1 } 0 } 1 transition. This test also covers 1).

Exceptions: Debug circuitry not testable
MCA bit 0 set from MPB
MAR bit 0 dump to MPB
MAR bits 3, 4, 5 dump to MPB not testable

3) All data cells tested for ability to hold a "0" and hold a "1" for 4096 states (refresh ability).

4) All multibit registers tested for adjacent bit independence (bit-wise R/W ability). This is to be accomplished by walking a "0" through a field of all ones, and a "1" through a field of all zeros. Note that a register that passes this test has also passed 2), and in fact most registers are only tested for R/W ability by this method. This test also insures adjacent bit independence of associated data paths (buses).

Exceptions: Sequencing stack registers other than uTR not testable. Word and page bus bit independence tested in module 11 (ROM checksum) as a side effect. uIR, uIR', and Micro-Instruction Bus not testable.

5) All micro orders tested for functionality with one case of remaining fields. i.e. Micro order independence of other micro orders in the same instruction is not practical to test.

Exceptions: CMAL, SMAL, HALT, POLLB, not testable. CUTRE, SUTRE, USTKP, DUPC, TRWB, TRNWB are debug only micro orders. HBDB5, PIN, PPOP are not testable by IOP alone, but are tested in system turn-on. NOP (all fields) not practical to test.

6) Test for one case of each microinstruction format for functionality of all micro order fields.

Exceptions: None.

7) Special hardware tested for functionality.

A) Combinatorial logic tested to insure each input signal can affect the output. This does not imply all input combinations are tested. For example, a three input NOR gate would be tested with an input sequence of 000, 100, 000, 010, 000, 001.

B) Incrementers tested for all combinations of carry in and bit contents for all bit positions.

C) Microcode transparent functions such as MPB hangs and interval timer updates are tested.

Exceptions: MPB address aborts never used.

8) ROM checksum test (Module 11)

A) Accumulated exclusive OR of all used ROM locations compared to a ROM constant.

B) Accumulated 2's complement addition of all ROM locations compared to a ROM constant (bits 0-5 separate from bits 6-37).

8.5 INDIVIDUAL MODULE DESCRIPTION

The following is a list of the self test modules with a short description of their intent. For more detailed information, see the IOP microcode listings.

1)IOP State Vector

This module tests most registers for stuck-at faults and refresh ability. It generates a progress pulse on debug pad 5, and is called as a subroutine in the beginning of each of modules 2 through 12. This module also leaves the IOP in a known state.

2) Loads, IODS, and Compare Tests

This module tests the functionality of LOAD, IODS, CMP8, CMP16, and CMP32. Also timing of CMP8, CMP16, and CMP32 is tested.

3) Counter Module

This module tests all non-sequencing stack incrementers.

4) Sequencing Module

This module tests special microinstruction sequencing and sequencing stack registers.

5) Register Read/Write Module

This module tests bit-wise read/write ability of most registers, through 0 and 1 pattern walks. The A Bus and Op-code Bus tested for bit independence as a side effect.

6) Bit-Set-Clear Module

This Module tests the functionality of GP1 shifts and the BSC logical functions AND, OR, and XNOR.

7) Formats Module

This module tests all microinstruction formats (See 8.5 #6). Also, output control lines (except RESET) are tested for R/W and refresh ability.

8) Tests and Specials Module

This module tests the following:

A) Functionality of TRIG, CBCRM, LL, NMA30, BYTE, IN, FIP, SETTD, CLRTD, PADMS, ITCU, CSTS, IODS, GPOS, and GPOTF.

B) Independence of TRIG, LL, BYTE, IN, and FIP.

9) Individual Tests

This module tests the ORCS and EWB microorders, the

priority encoder, and the dump constant hardware.

10) I/O Bus Hardware

This module tests the data paths to the I/O bus pads, input control lines and PAIC output pads for stuck at faults. Reset line tested for refresh.

11) Control Store Check

ROM checksum words are computed and compared to ROM constants (See 8.4 #8).

12) MPB Internal

This module tests bit-wise read/write ability of the MAR register. No MPB transactions occur.

13) MPB Channel to Channel Tests

This module tests MPB registers for refresh ability and stuck at faults, and tests non-memory access MPB functions.

14) MPB Memory

This module tests all microorders associated with accessing memory for functionality.

9. HARDWARE DESCRIPTION

The hardware pieces of IOP can be broken into eight parts as shown in Figure 9.1. These parts are: Registers and Special Functions, DMA Registers, Micro-Sequencer, PLA Microinstruction Decoder, ROM Control Store, MPB Control and Interface, and I/O Bus Interface section. Each will be described in the following sections.

9.1 REGISTERS AND SPECIAL FUNCTIONS

This section describes the registers involved in Interrupt, Micro-Instruction, I/O Bus, Interval Timer, and other general purpose registers. The Special Functions include all data manipulation and data comparing.

9.1.1 Interrupt

The registers used in connection with interrupt functions are the Interrupt Request register (IRQ), the Interrupt Message register (IMSG), and the Interrupt Mask register (IMSK). (See Figure 9.2).

The Interrupt Request register is an eight bit register which can be set from or dump to the least significant eight bits of the A Bus. A one in a bit position corresponding to an interface indicates an interrupt is requested for that interface.

The Interrupt Message register is a 32 bit register which can be set from or dump to the A Bus. IMSG(3:3) contains the channel number of the CPU that services the interrupts from the IOP. If it is zero, a broadcast message is issued instead of a channel message. The lower 8 bits are used by IOP as a mask.

The Interrupt Mask register is an 8 bit register which contains the mask bits to enable interrupt for each PA. The IMSK can be set from or dump to the A Bus (24:8).

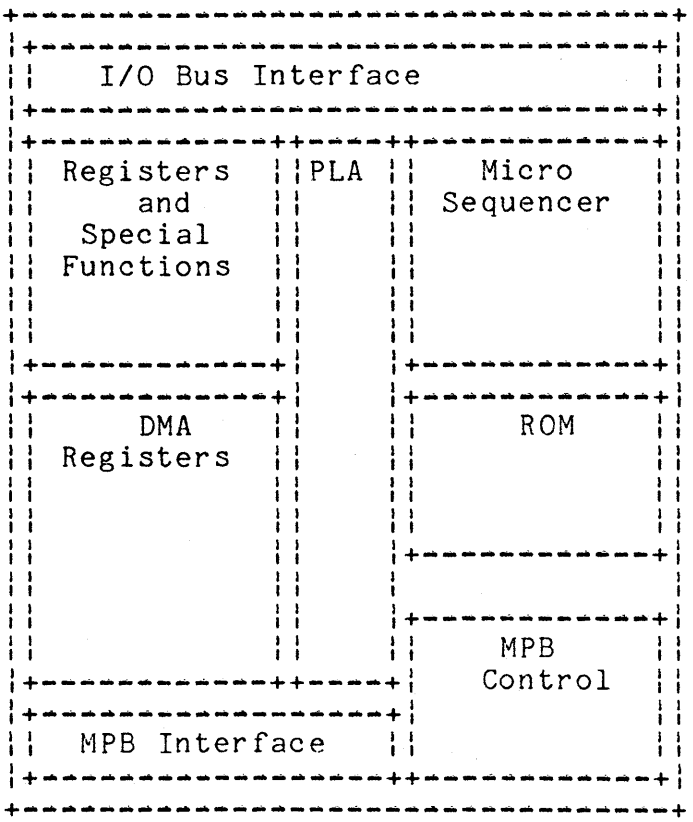


Figure 9.1 IOP Functional Blocks

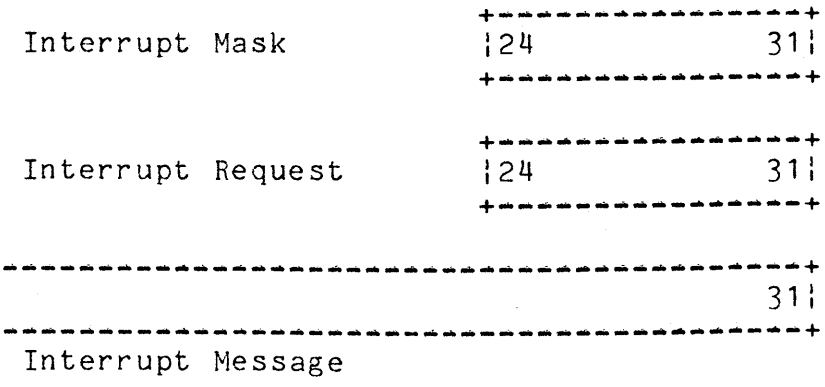


Figure 9.2 Interrupt Registers

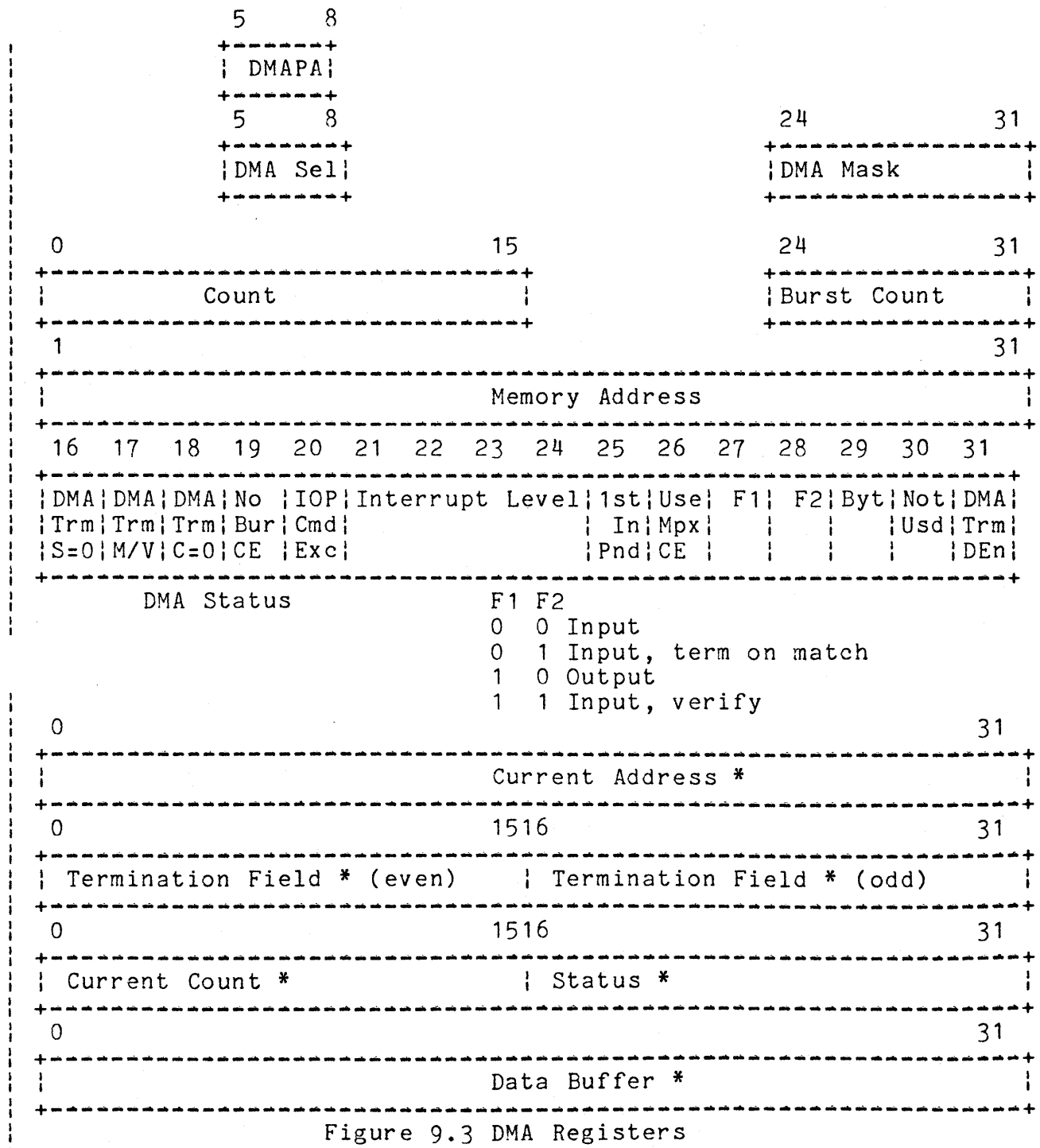


Figure 9.3 DMA Registers

				MODEL	STK #
				I/O System ERS	
				BY Fred Gross	DATE 09/20/82
LT	P.C. #	APPR	DATE	APPD	SHEET # 55 OF 80
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

9.1.2 Micro-Control Registers

The Micro Instruction Register (uIR) consists of a 38 bit uIR and 12 bit LuIR. The uIR can be set from or dump to the 38 bit Micro Instruction Bus (uIB). The uIR register can also be shifted left or right one bit with the B Bus bit 31 (normally zero) shifted in when shifting left with the output going external and an external input is shifted in when shifting right. The LuIR register can be set from the uIR (0:6,19:6) or dump to the uIB (0:6,19:6).

The control store data can be read by a Read Control Store (RCS) instruction. The lower 32 bits are loaded into GP0 and the upper 6 bits are loaded into GP1(0:6).

The Micro Trace register is a 14 bit register which can be set from or dump to the IOD register (2:14) via the opcode bus and can be set from the Page Bus or the Word Buses.

9.1.3 I/O Bus Registers

The Input/Output Data (IOD) register is a 32 bit register which can be set from or dump to the A-Bus or the least significant 16 bits can be set from the external I/O bus without disturbing the upper 16 bits. The upper 16 bits drive the IOD pads through buffers. When loading the IOD register from the A-Bus, data comes true at the pads at the end of the phase two clock in the state following the instruction state. The upper half-word and the lower half-word can be exchanged. The upper half, IOD (0:16) can be loaded from or dumped to the Opcode Bus.

The Peripheral Address/Interface Control (PAIC) is a seven bit register which can be set from or dump to bits 6 to 12 of the A-Bus.

Figure 9.4 shows the correspondence between the IOD register and the external I/O bus.

9.1.4 Interval Timer Registers

The Timer Start (TS) register is a 32 bit register which can be set from or dump to the A-Bus. The Timer Count (TC) register is a 32 bit register which can be set from or dump to

the A-Bus. The lower 8 bits retain their previous value when setting from the A Bus due to a timer underflow (See sec. 7.2). The Load Timer Count command loads all bits.

9.1.5 Other Registers and Functions

The 32 bit General Purpose #0 (GP0) register can be set from or dump to the A-Bus. The upper 16 bits can be swapped with the lower 16 bits. The 32 bit General Purpose #1 (GP1) register can be set from the A-Bus or bits 0 to 5 (0:6) can be set from or dump to the most significant six bits of the B Bus. It can be shifted left one bit with B Bus bit 31 shifted in (i.e. DB5; with no external connection to DB5, a zero will be shifted in) or right one bit with a 0 shifted in.

A Compare (COMP) function which compares the four bytes of GP0 with the four bytes of IOD, signals the result with four lines, each line going true if the two bytes match. Compares can be done on 32 bits, 16 least-significant-bits, or 8 least-significant-bits.

A Bit Set/Clear (BSC) function performs logical operations between GP0 and IOD and places the result on the A-Bus. The logical operations are: OR, AND, and XNOR. The BSC outputs appear in the next state and can be used any number of states later (static).

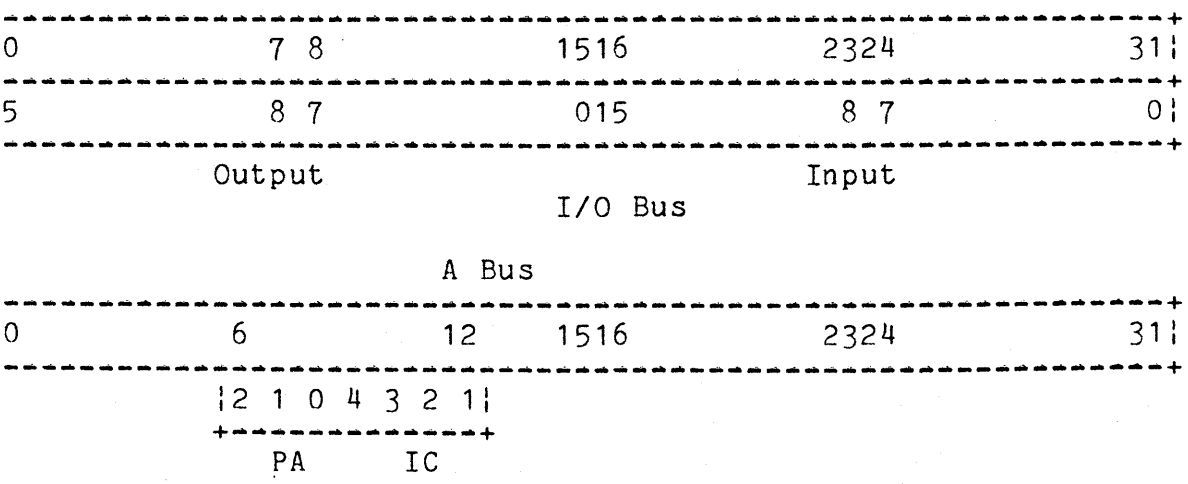


Figure 9.4 External Bit Numbering to Internal

A Constant (CON) function will set any of the least significant sixteen bits of the A Bus (16:16). The DMS register points to the bit set. One microcode state must be allowed between loading DMS and dumping the constant.

9.2 DMA REGISTERS

The DMA Registers consist of a group of active registers involved with the DMA currently active on the I/O Bus and eight sets of logical registers which hold information until the interface to which they correspond becomes the active DMA. (See Figure 9.3)

9.2.1 DMA Active Registers

The DMA Mask (DMSK) register is an eight bit register which can be set from or dump to the A-Bus (24:8). A one enables a PA for DMA.

The Count (C) register is a sixteen bit register which can be set from or dump to the A-Bus (0:16). It can be decremented and it sets a qualifier true when the count goes negative.

The Burst Count (BC) register is an eight bit register which can be preset to thirty-one or be decremented and it sets a qualifier true when the count goes negative.

The Memory Address (MA) register is a thirty-two bit register which can be set from or dump to the A-Bus and it can be incremented. A bit, MA (30:1), is used as a halfword or word address qualifier.

The Status (STS) register is a sixteen bit register which can be set from or dump to the A-Bus (16:16). In addition, five bits are used as qualifiers.

The DMA Select (DMS) register is a four bit register which can be set from or dump to the A-Bus (5:4). The DMS register points to the register accessed when transferring between the A Bus and any DMA logical register. The most significant bit is ignored when addressing the registers.

The DMAPA register is a four bit register which can be set from or dump to the A-Bus (5:4). The DMAPA register (6:3) points to the PA enabled to use the RAMD line, with bit (5:1) used to enable RAMD.

9.2.2 DMA Logical Registers

There are eight of each of the following registers numbered zero to seven with the number designation following the mnemonic when the register is referenced. The number corresponds to a peripheral address.

The Current Address (CA*) register, the Termination Field (TF*) register, the Current Count/Status (CCST*), and the Data Buffer (DB*) register are all thirty-two bit registers which can be set from or dump to the A-Bus. There are only four TF* registers which are shared by two numerically adjacent PA's, the even in bits 0-15, odd in bits 16-31.

Since only byte match DMA is allowed, there are eight bytes in the TF* registers that are not used. Currently, bits 0-7 of TF0 are used as the Attention Poll mask and bits 16-23 are used as an Attention Acknowledge indicator (see CSG I/O Standards).

MICROPROGRAM SEQUENCING AND SUPPORT HARDWARE

The flow of microcode from the control store to the microinstruction decoders is controlled by the microsequencing hardware. Several 14 bit registers and associated incrementers contain or compute the microcode addresses. Branch conditions combine with the current microinstruction to determine which register or incrementer will provide the address of the next microinstruction. See Figure 9.5 for a diagram of the sequencing hardware.

9.3.1 Addressing Control Store

Control Store addresses are composed of two parts, page address and word address, sent on separate buses. The Page Bus is a 14 bit bus and the Word Bus is a 6 bit bus with both positive and negative true data. Another bus called the Opcode Bus ties the sequencing hardware to the IOD register.

9.3.2 Microaddress Registers and Incrementers

The microcode program counter, uPC, normally contains the address of the next microinstruction to be fetched from control store. Tied to the uPC is a three deep subroutine return stack where microcode return addresses are stored and an incrementer to provide the next microinstruction address.

The target address of a microcode jump is held in the microjump register, uJR. This register is loaded on the state preceeding the execution of the microinstruction containing the jump.

Traps cause the control store address to come from one of the 2 locations in the trap address ROM. Traps also cause the uPC to be pushed onto the microsubroutine stack.

The 14 bit microtrace register, uTR, provides a simple breakpoint capability to aid in debugging IOP hardware and microcode. This register causes the execution of microinstructions to be halted when the control store page bus matches its contents.

9.3.3 Special Sequencing

If a microinstruction causes an off-page branch which empties the microinstruction pipeline, then the microsequencing state machine provides a NOP state while the pipe is refilled. Some operations causing NOP states are: long jumps, long subroutine jumps, and traps.

9.3.4 Microcoding IOP with Pipelined Microcode

IOP microcode is pipelined. Different instructions are fetched, decoded and executed at the same time. Microinstructions can be effectively executed at the rate of one every state time, but a microinstruction may be in the pipeline for several states. The longest pipeline sequence occurs for I/O operations.

For I/O operations (Set/Clr IOSB, DOUT, POUT, INT, RST) if the instruction is fetched during a certain clock phase one, the signal will be true at the pad by the end of the second clock phase two following the instruction fetch. It will be true at the interface 25ns later.

Another place where the pipeline sequence of microcode has to be taken into account is when the "status register" is loaded and a jump taken on the condition of a bit in the "status register".

9.4 PLA MICROINSTRUCTION DECODER

Microinstructions on the Microinstruction Bus are sent to PLA drivers and the PLA's. Control line outputs from the PLA set, dump, and perform other functions throughout IOP, according to the microinstruction definition (see I/O Microcode ERS). During clock phase two, all control line outputs are pulled low (false).

9.5 ROM CONTROL STORE

ROM Control Store consists of 4608 words by 38 bits of read-only-memory. The address of fourteen bits (one is unused) comes in two fields, a ROM page address of nine bits and a ROM word address of five bits. The ROM outputs drive the Microinstruction Bus.

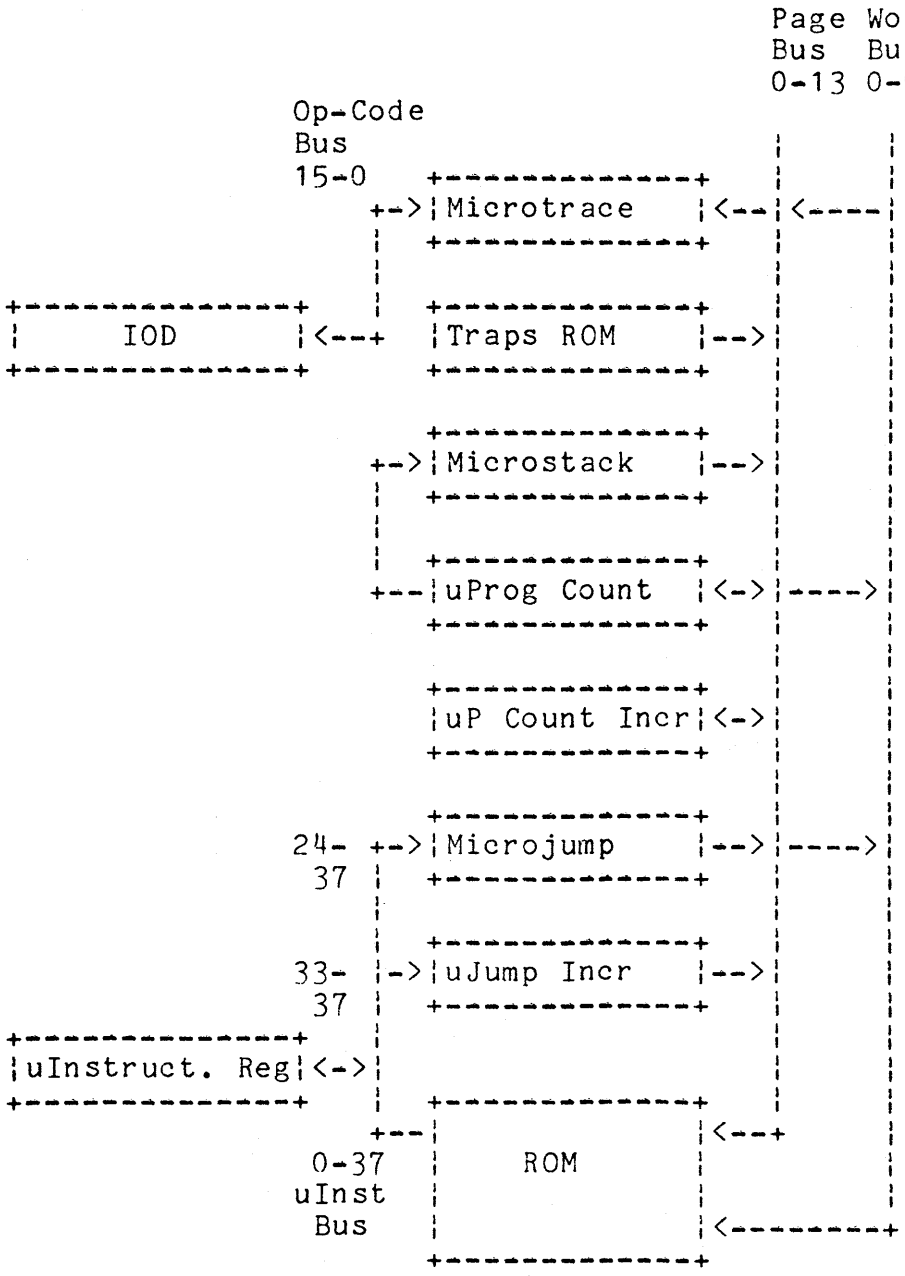
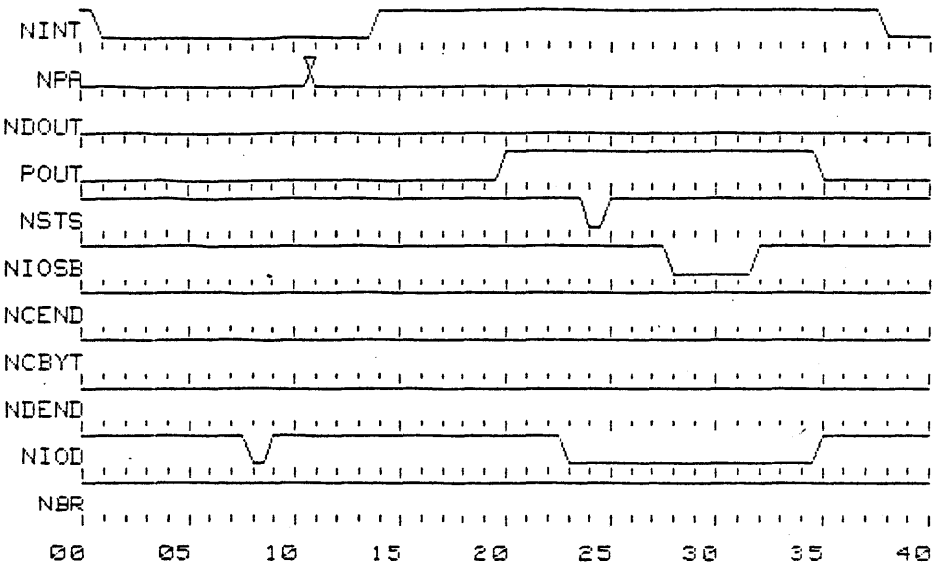


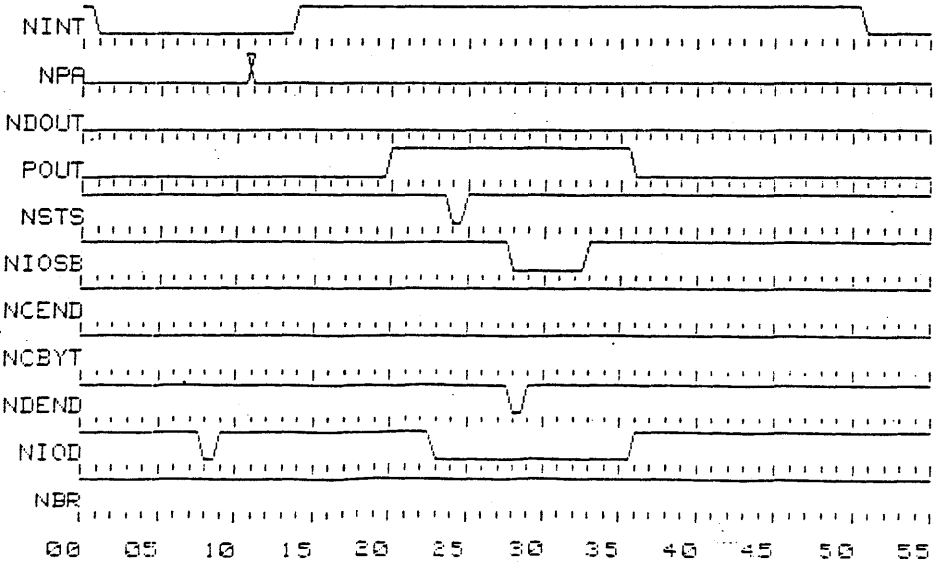
Figure 9.5 Microcode Sequencing Hardware

HEWLETT-PACKARD CO.

OUTPUT HALFWORD MULTIPLEX



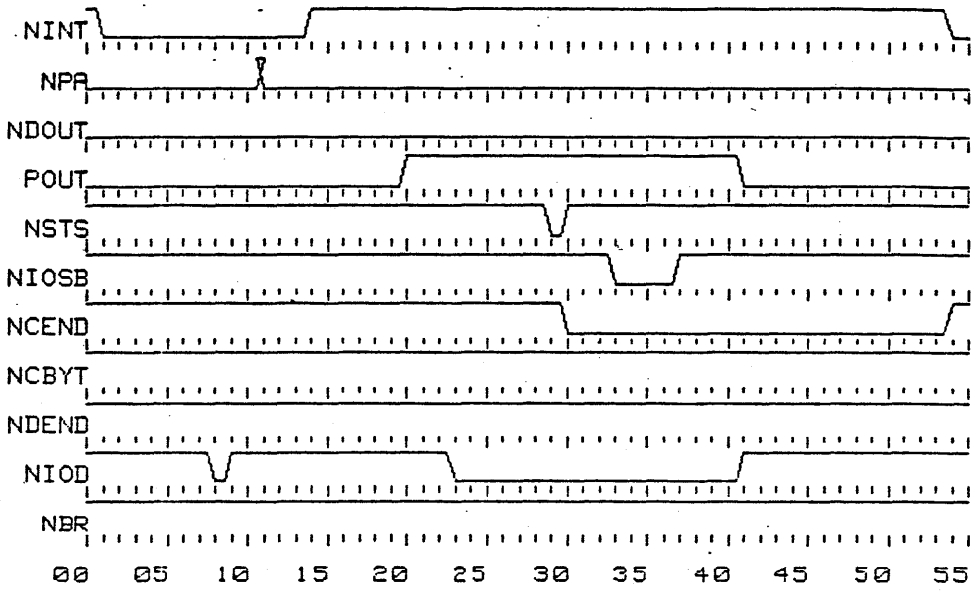
OUTPUT HALFWORD MULTIPLEX - DEVICE END



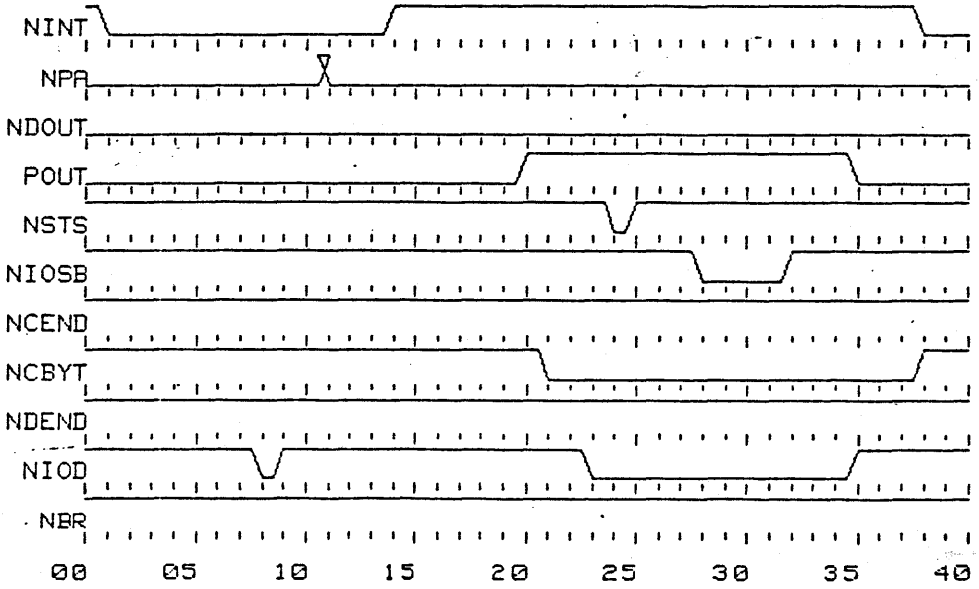
			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR	DATE	APPD	SHEET # 63 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

HEWLETT-PACKARD CO.

OUTPUT HALFWORD MULTIPLEX - CHANNEL END

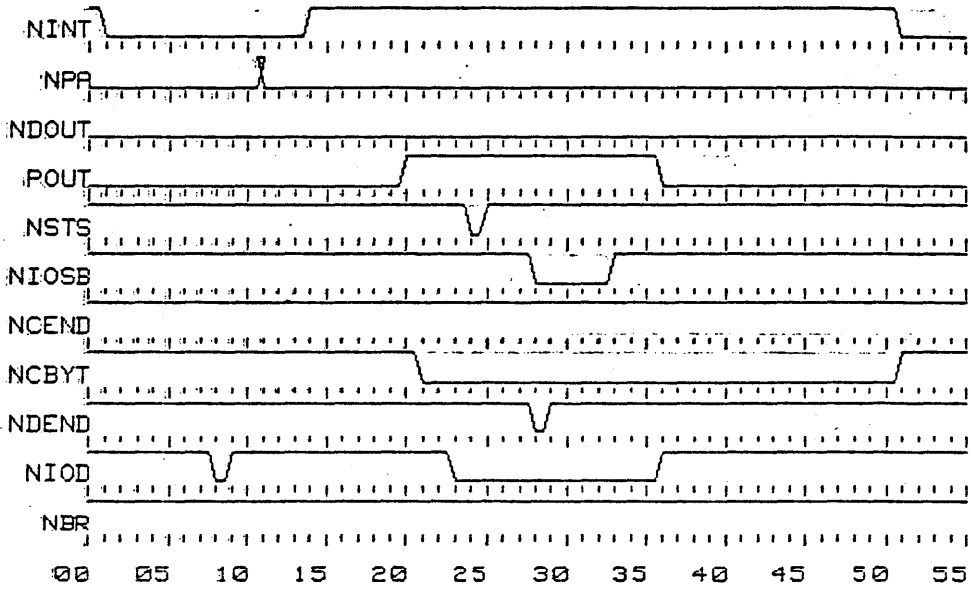


OUTPUT BYTE MULTIPLEX

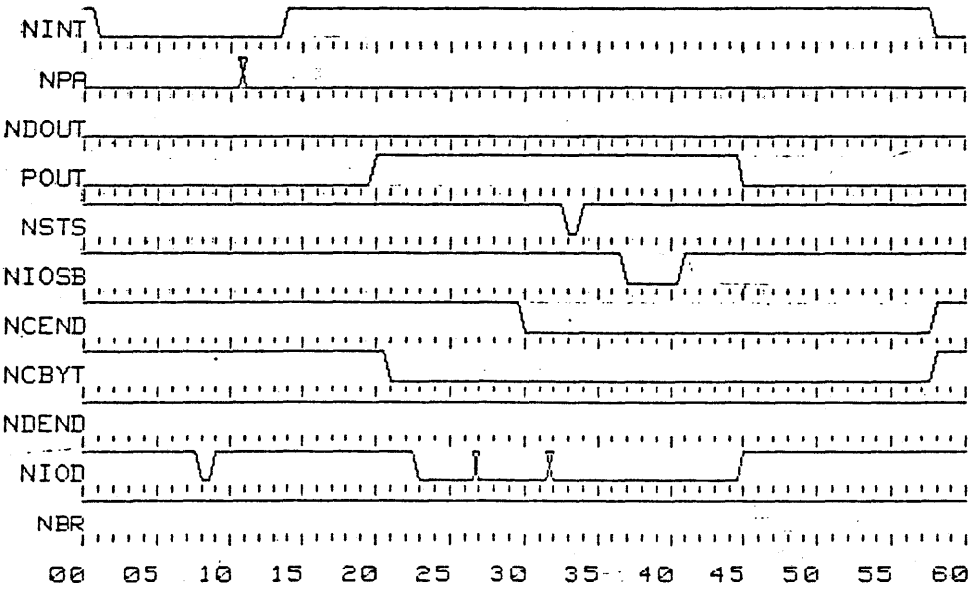


			MODEL	STK #	
			I/O System ERS		
			BY Fred Gross	DATE 09/20/82	
LT	P.C. #	APPR	DATE	APPD	SHEET # 64 OF 81
	REVISIONS		SUPERSEDES		DWG # A-1FE1-3030-9

OUTPUT BYTE MULTIPLEX - DEVICE END

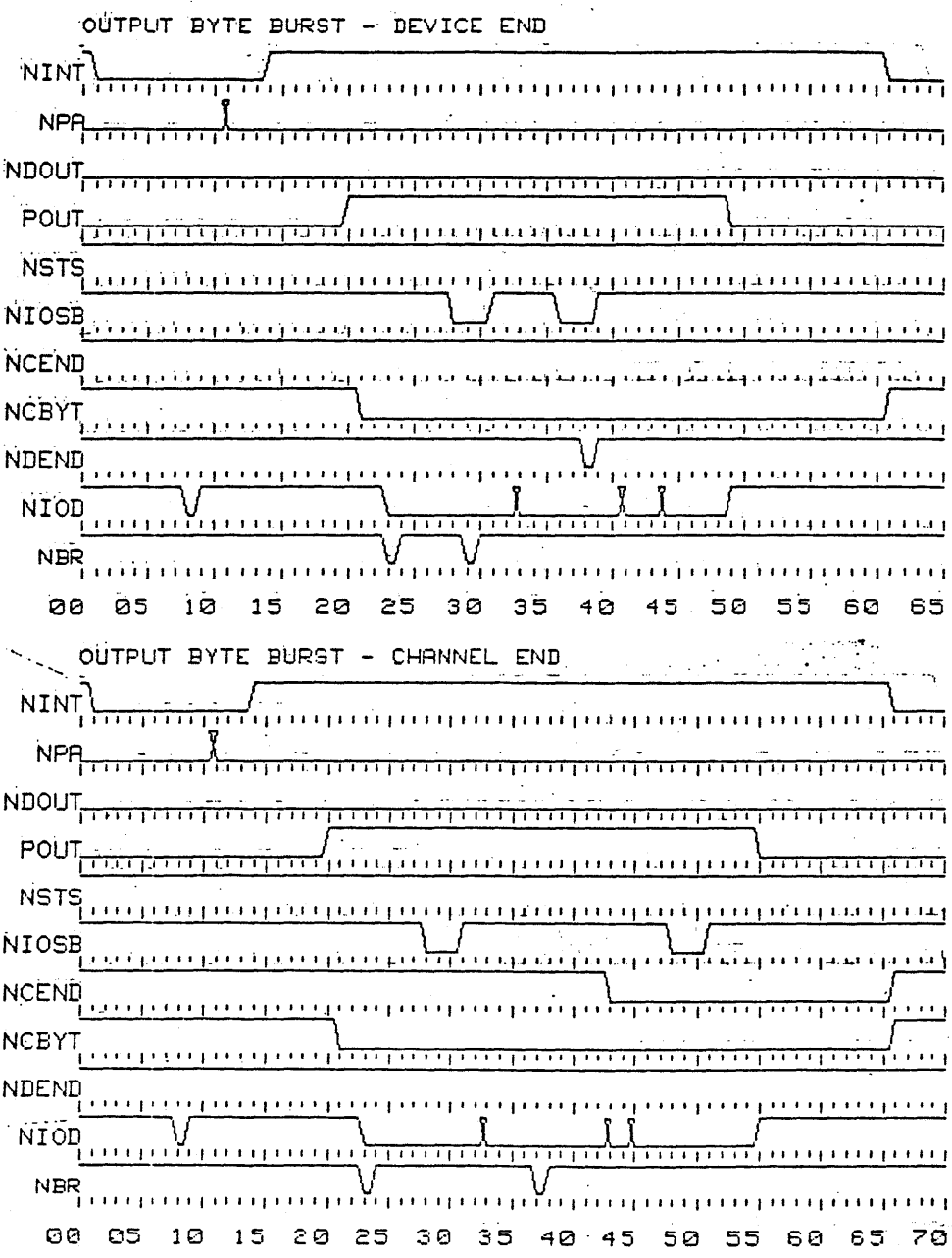


OUTPUT BYTE MULTIPLEX - CHANNEL END



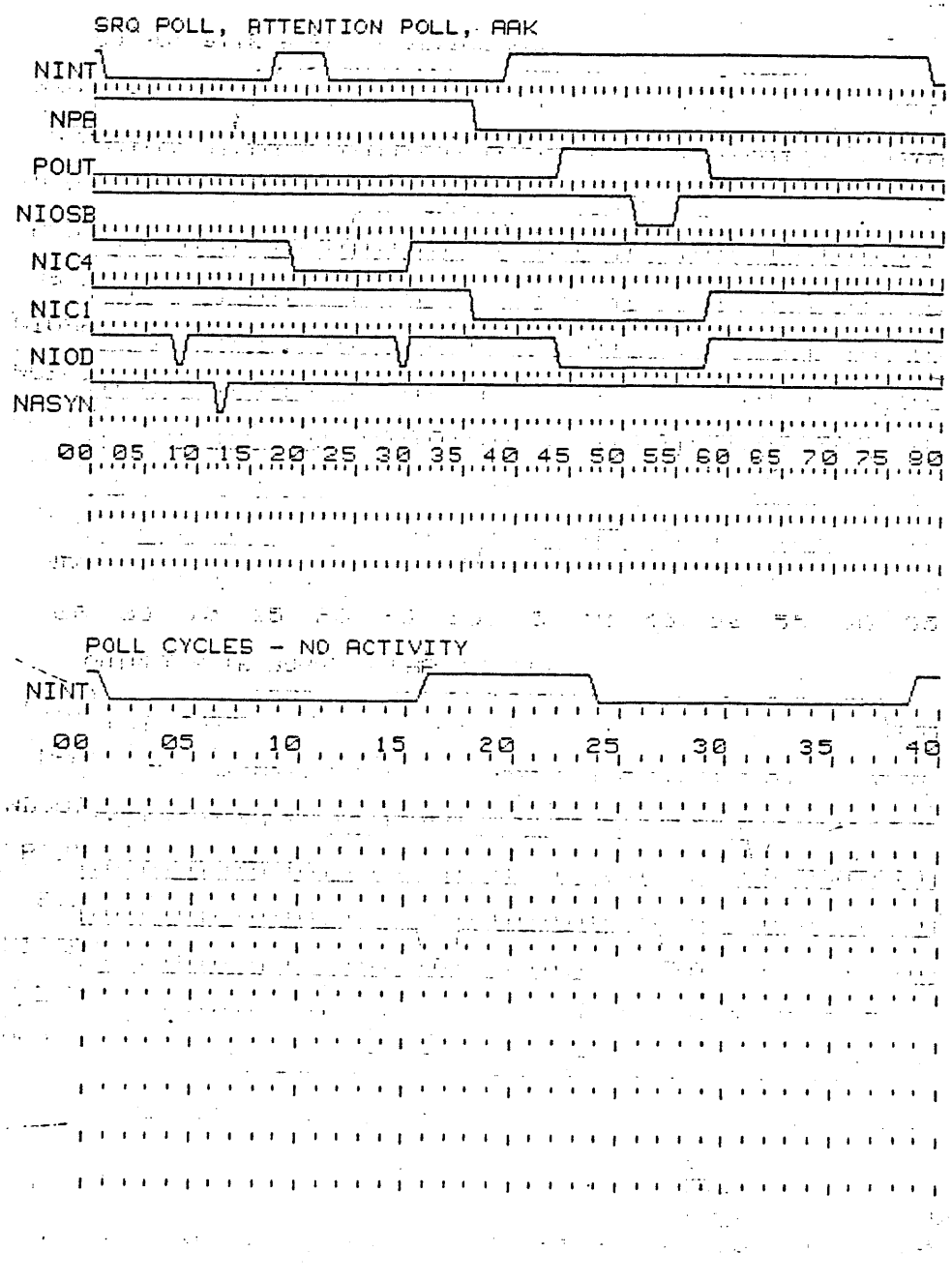
			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR DATE	APPD	SHEET # 65 OF 81
	REVISIONS		SUPERSEDES	DWG # A-1FE1-3030-9

HEWLETT-PACKARD CO.

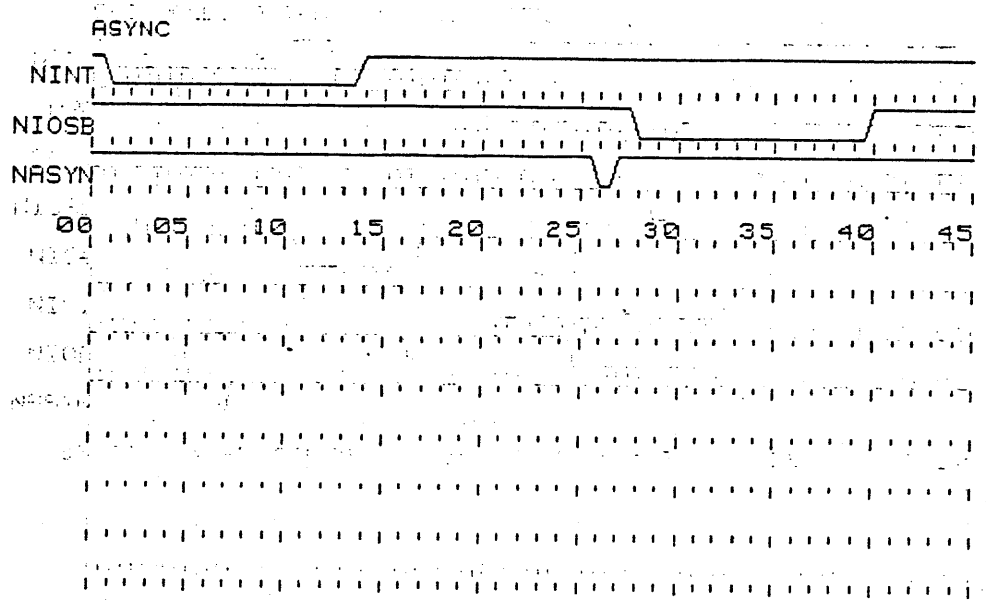


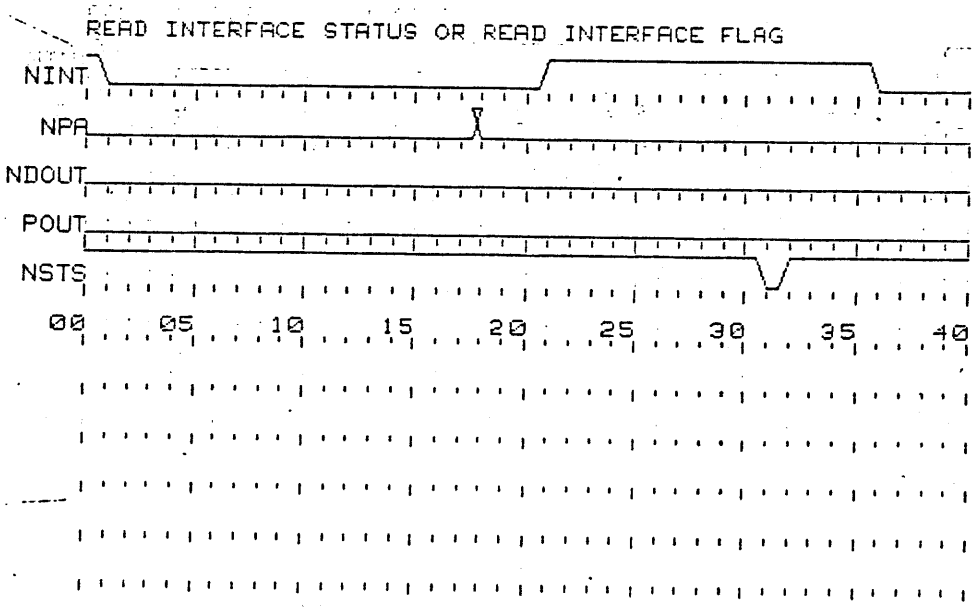
			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
DI	P.C. #	APPR	DATE	APPD	SHEET # 66 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

HEWLETT-PACKARD CO.



			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR	DATE	APPD	SHEET # 67 OF 81
	REVISIONS		SUPERSEDES	DWG # A-1FE1-3030-9

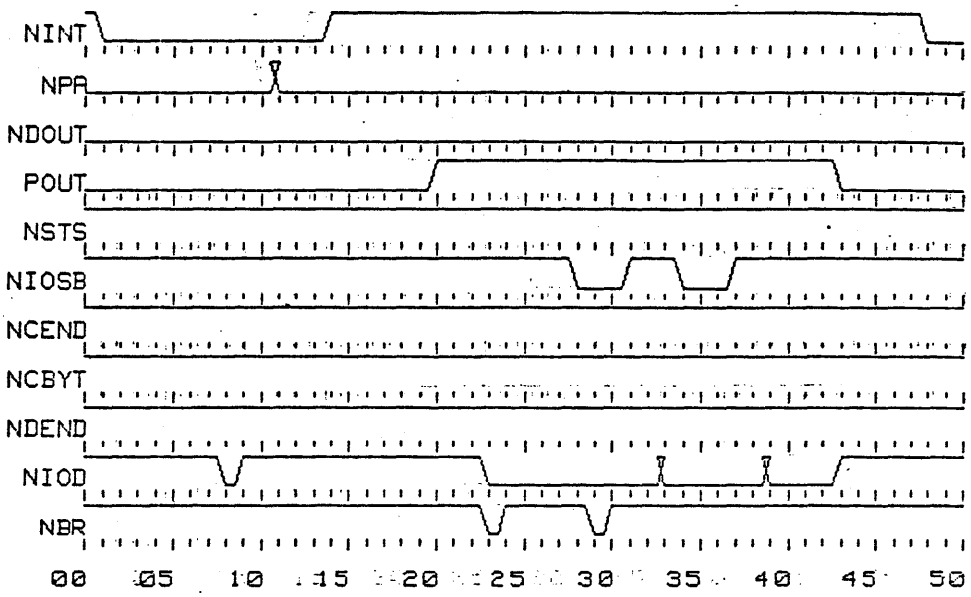




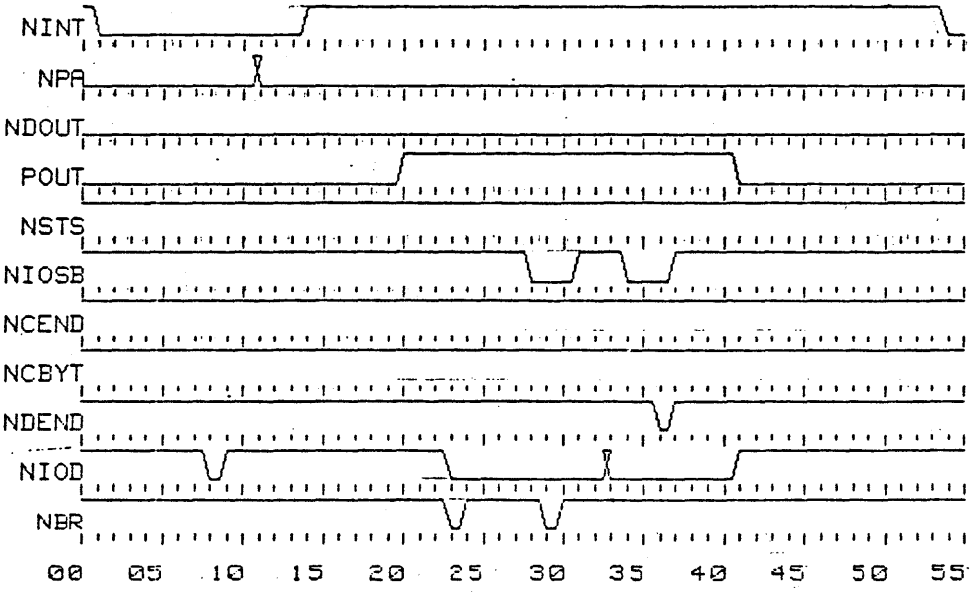
			MODEL	STK #	
			I/O System ERS		
			BY Fred Gross	DATE 09/20/82	
LTR	P.C. #	APPR	DATE	APPD	SHEET # 68 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

HEWLETT-PACKARD CO.

OUTPUT HALFWORD BURST

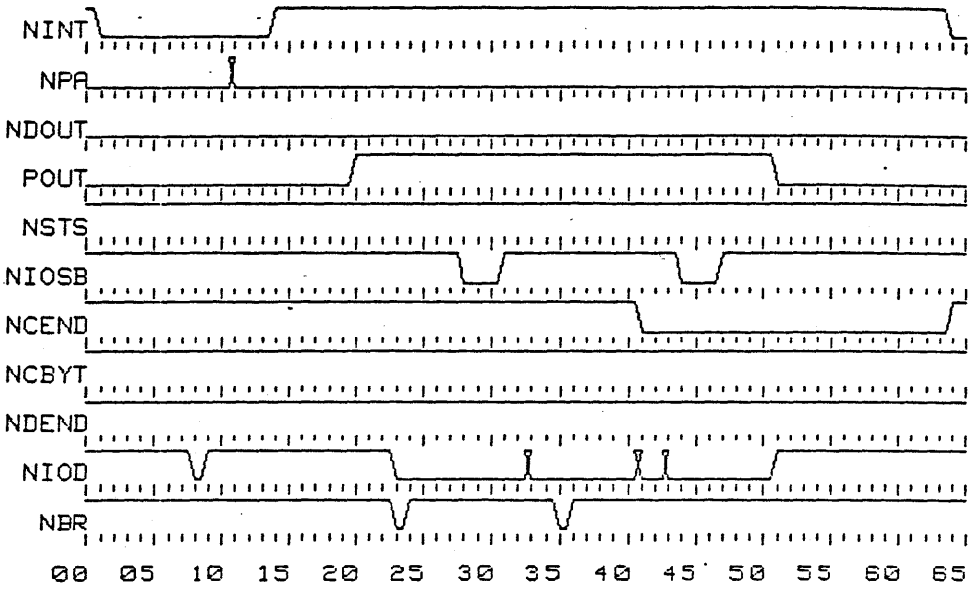


OUTPUT HALFWORD BURST - DEVICE END

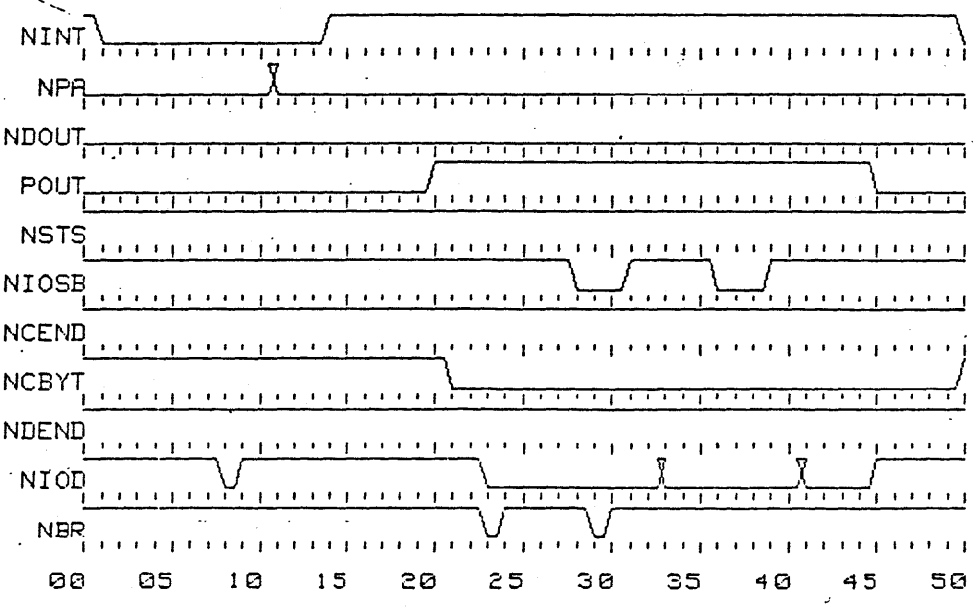


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR	DATE APPD	SHEET # 69 OF 81
REVISIONS			SUPERSEDES	DWG # A-1FE1-3030-9

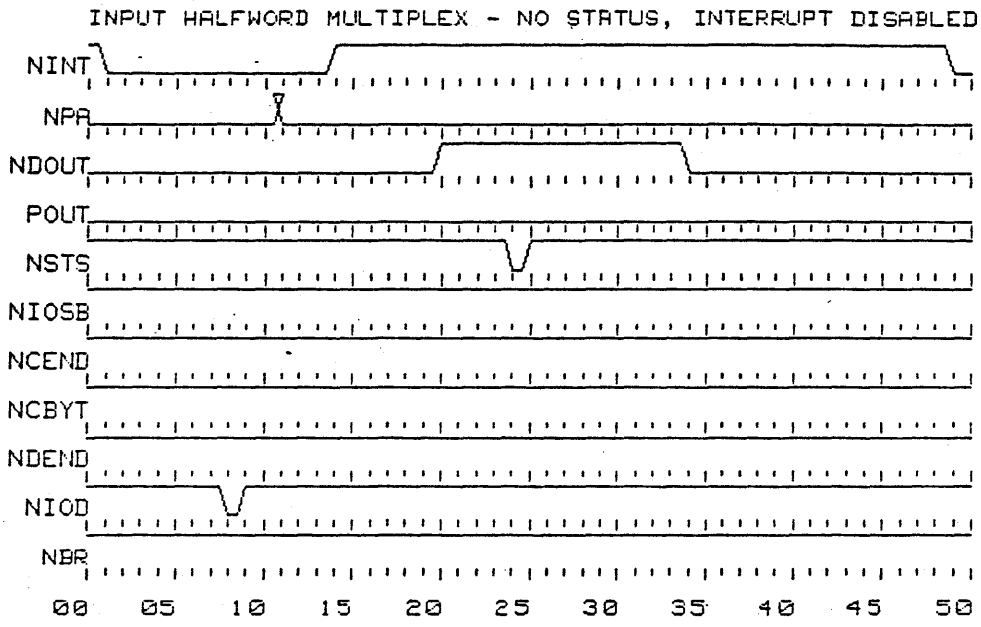
OUTPUT HALFWORD BURST - CHANNEL END



OUTPUT BYTE BURST

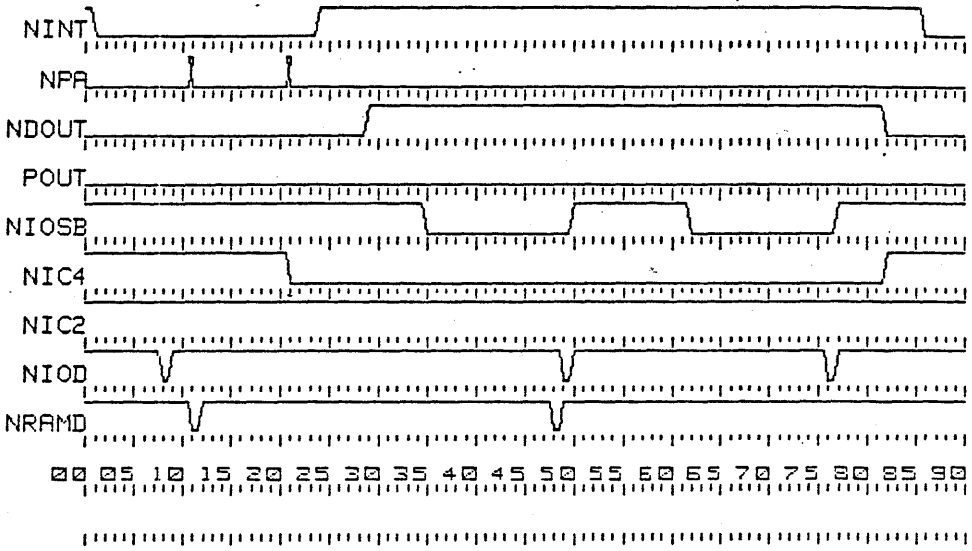


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LT	P.C. #	APPR	DATE	APPD	SHEET # 70 OF 81
	REVISIONS		SUPERSEDES	DWG # A-1FE1-3030-9

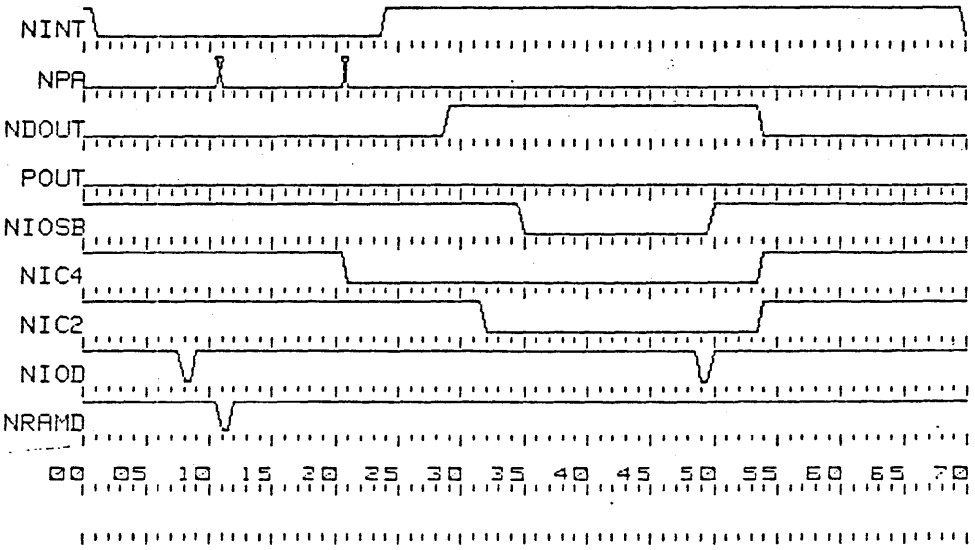


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR	DATE	APPD	SHEET # 71 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

INPUT RAMD

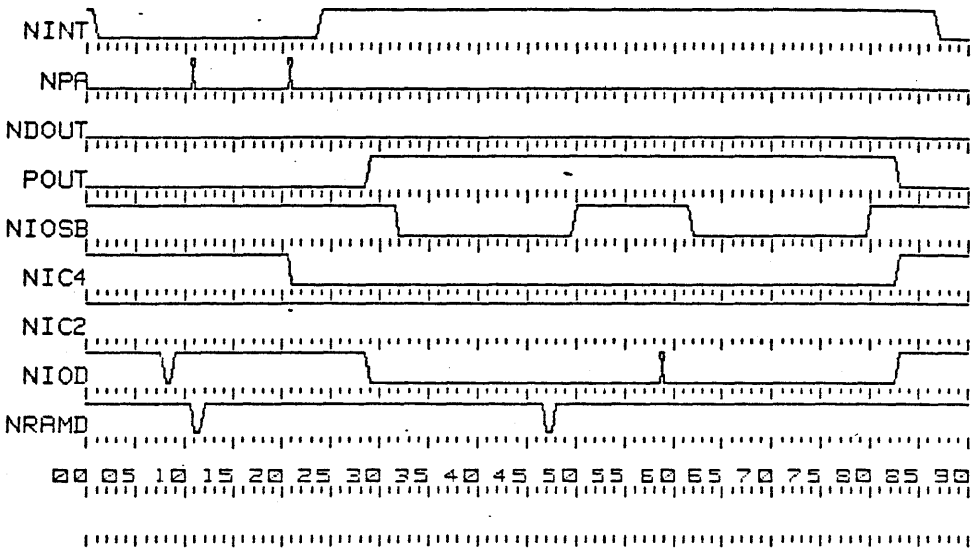


INPUT RAMD - LAST TRANSFER

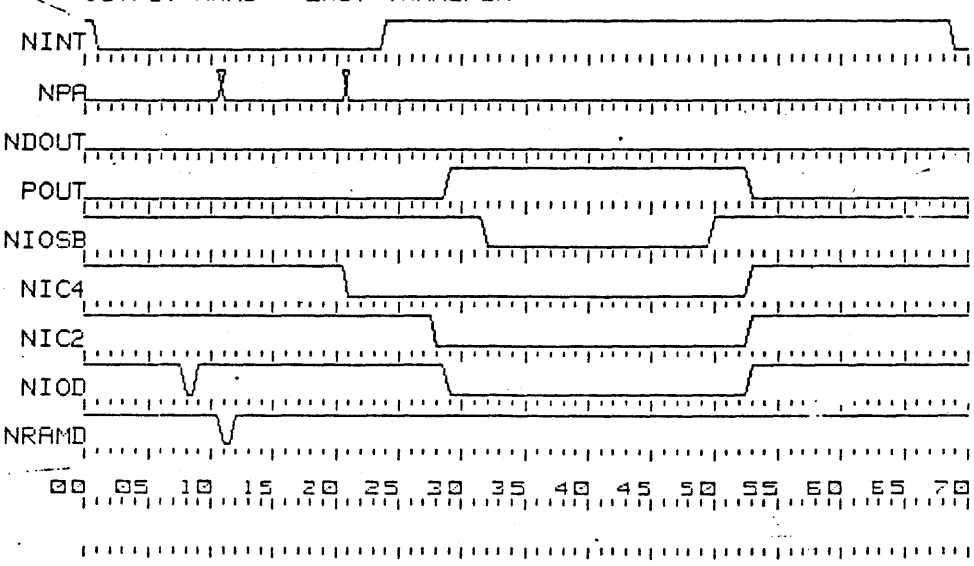


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR	DATE	APPD	SHEET # 72 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

OUTPUT RAMD

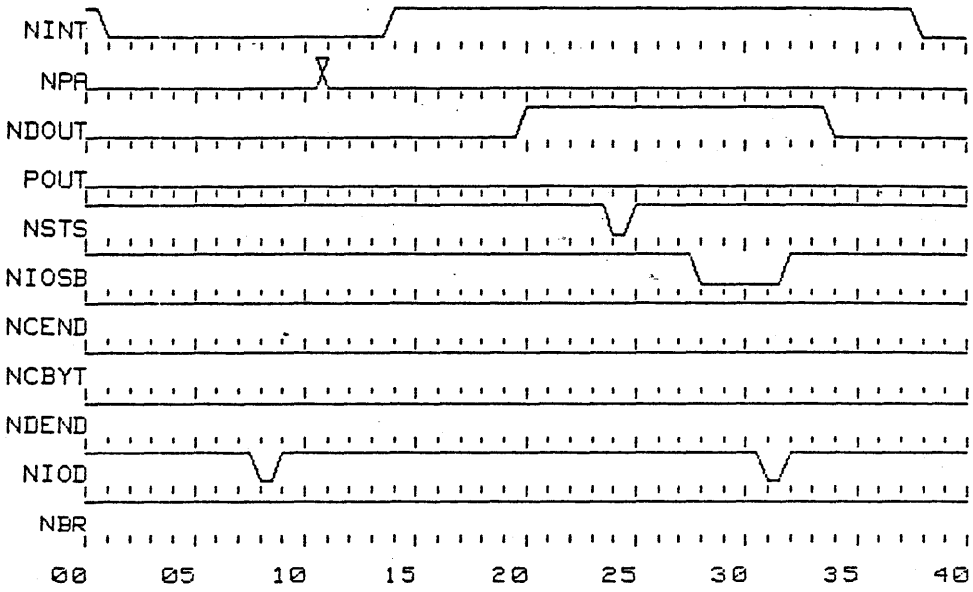


OUTPUT RAMD - LAST TRANSFER

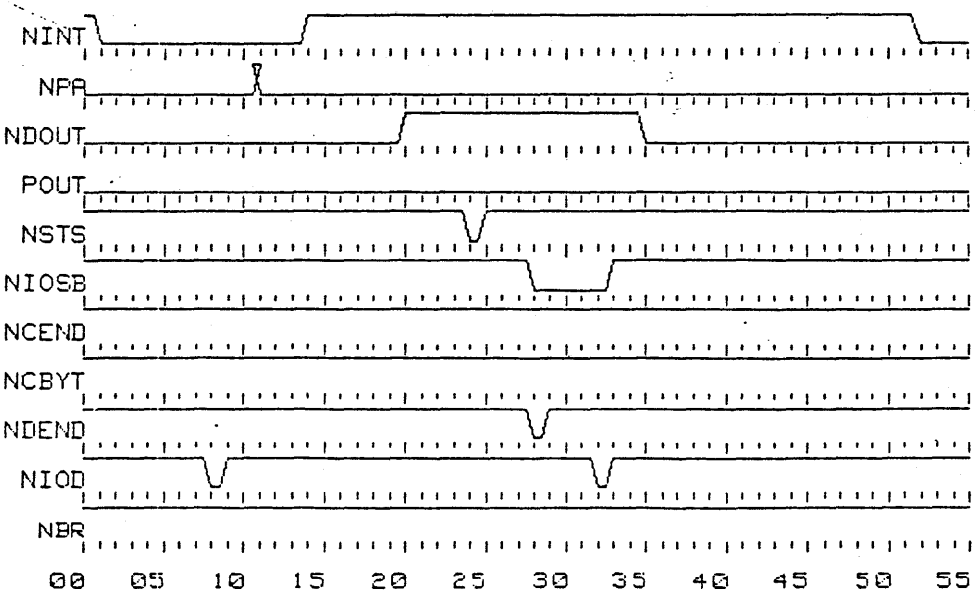


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
DT	P.C. #	APPR	DATE	APPD
				SHEET # 73 OF 81
	REVISIONS		SUPERSEDES	DWG # A-1FE1-3030-9

INPUT HALFWORD MULTIPLEX

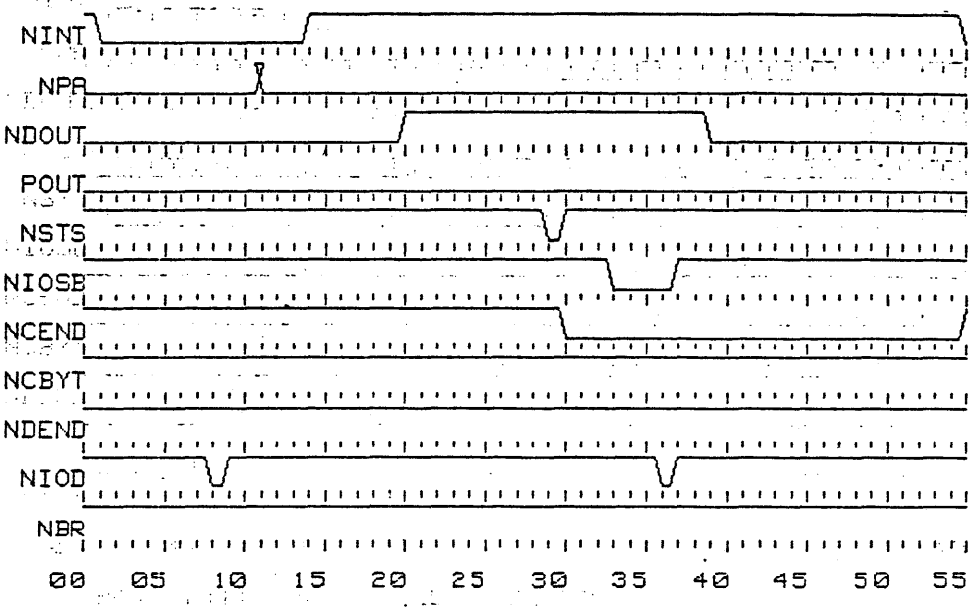


INPUT HALFWORD MULTIPLEX - DEVICE END

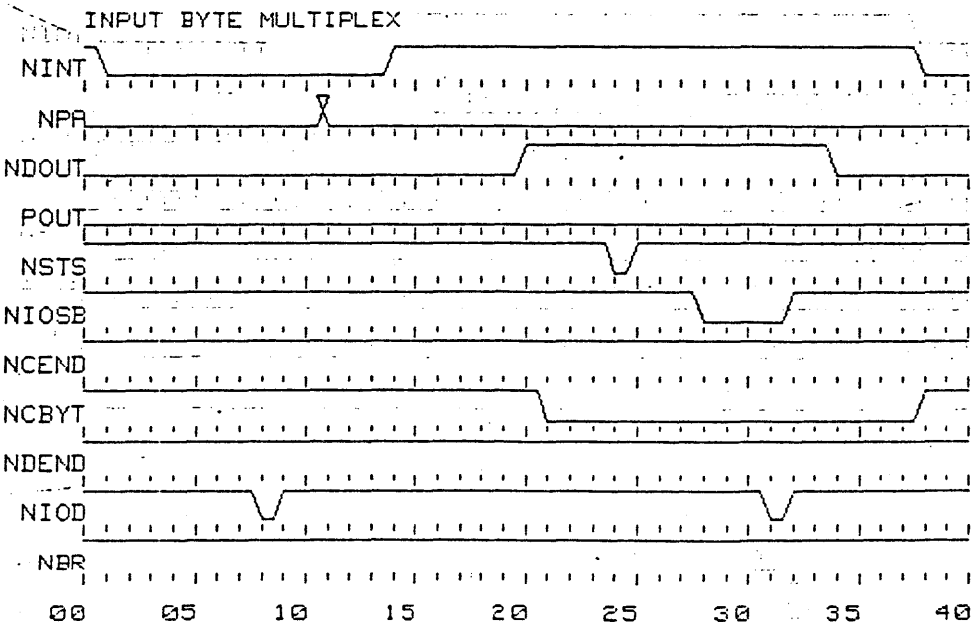


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR	DATE	APPD	SHEET # 74 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

INPUT HALFWORD MULTIPLEX - CHANNEL END



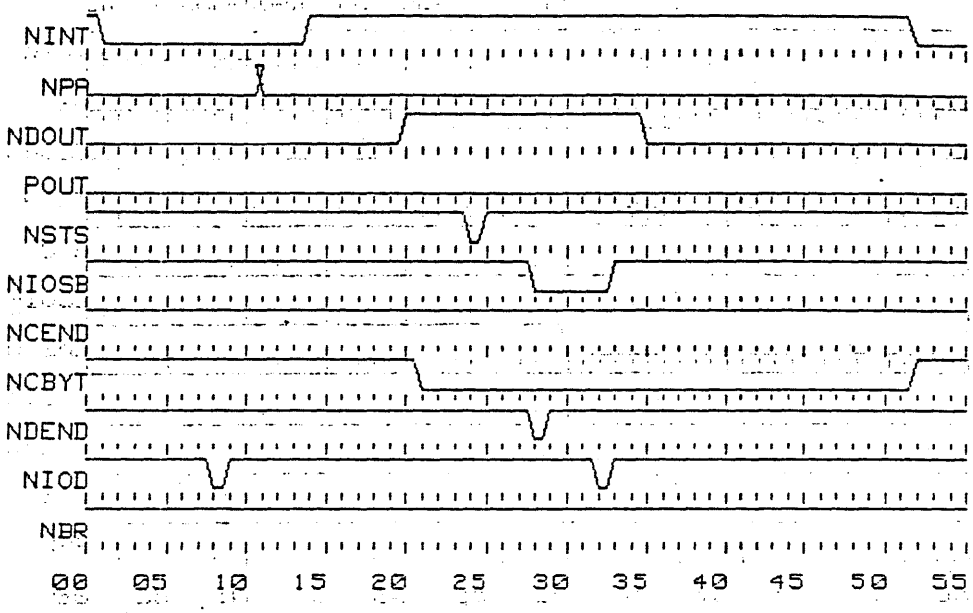
INPUT BYTE MULTIPLEX



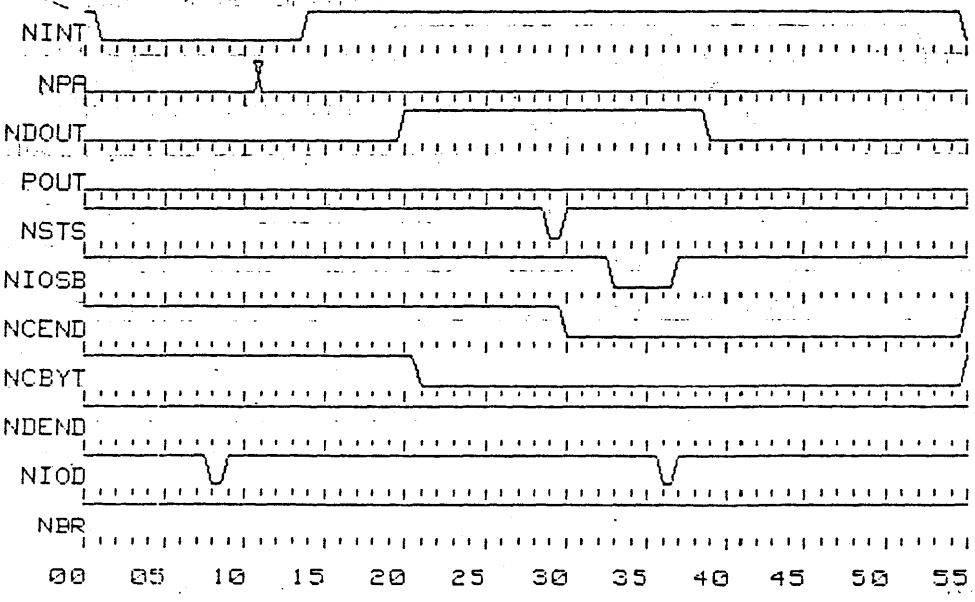
			MODEL	STK #	
			I/O System ERS		
			BY Fred Gross	DATE 09/20/82	
LTR	P.C. #	APPR	DATE	APPD	SHEET # 75 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

HEWLETT-PACKARD CO.

INPUT BYTE MULTIPLEX – DEVICE END

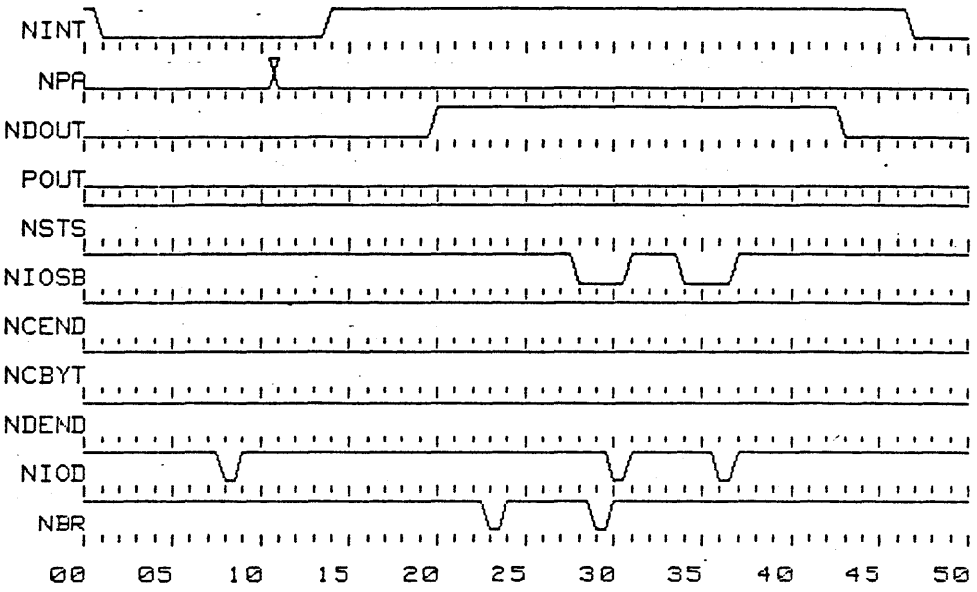


INPUT BYTE MULTIPLEX – CHANNEL END

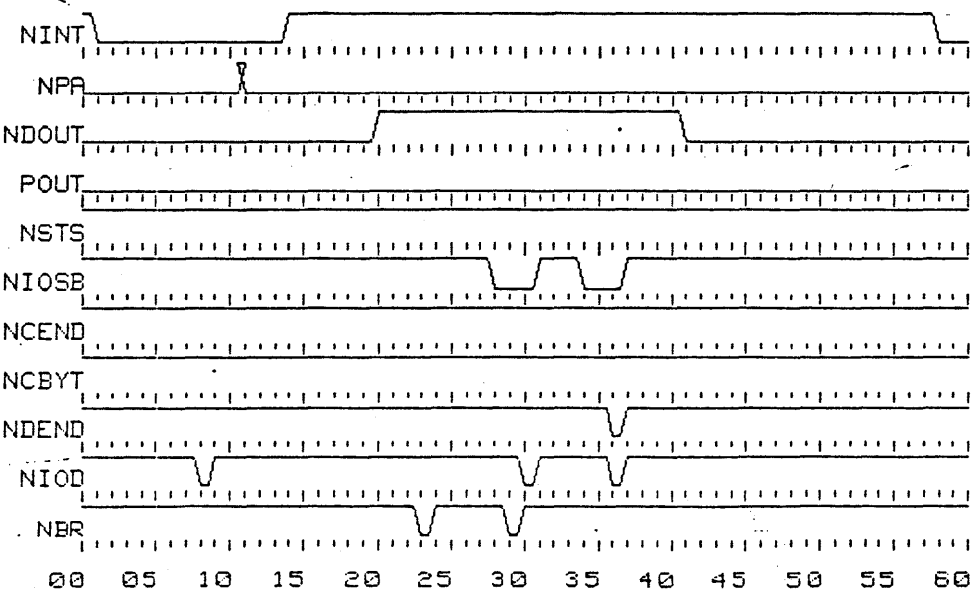


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
NO.	P.C. #	APPR	DATE	APPD	SHEET # 76 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

INPUT HALFWORD BURST



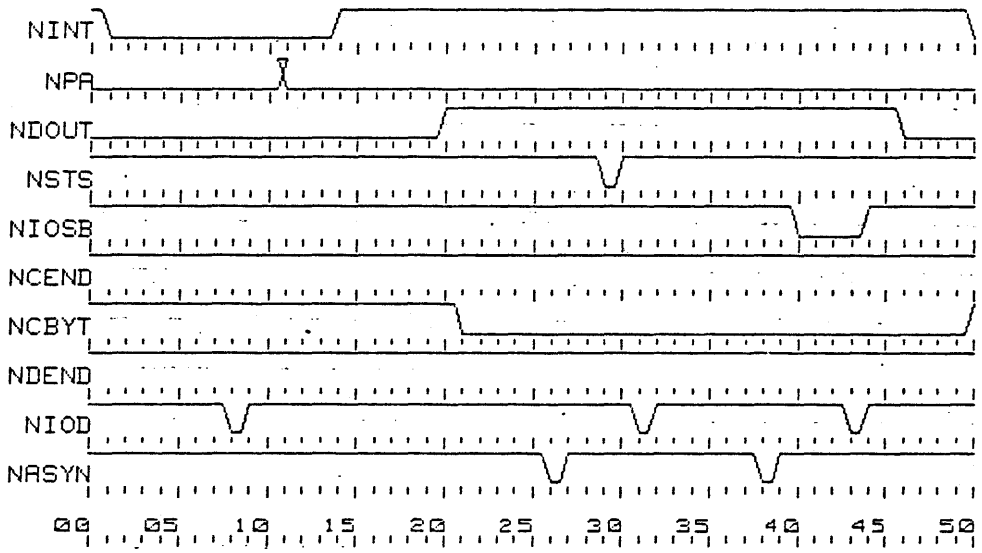
INPUT HALFWORD BURST - DEVICE END



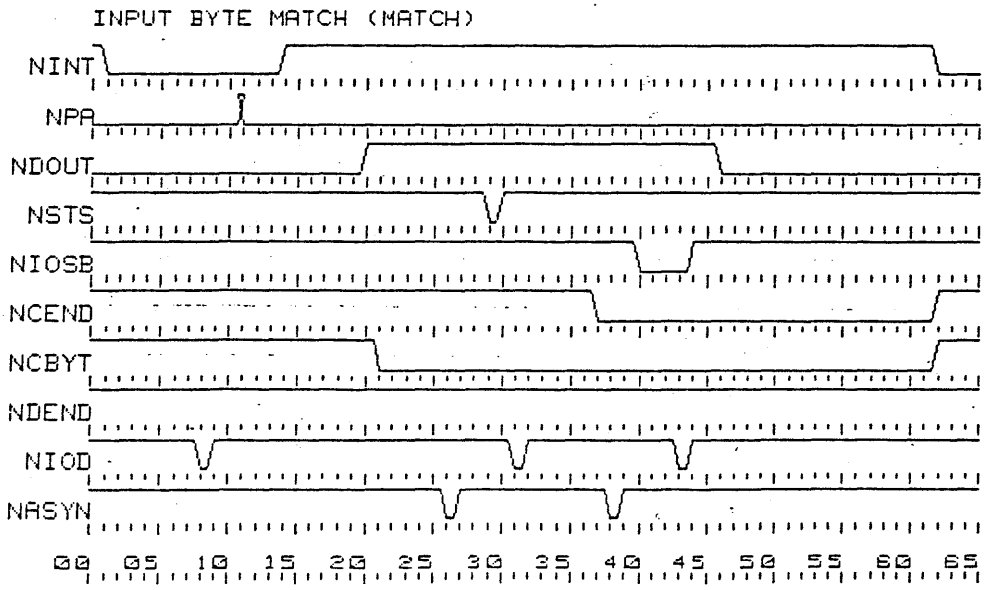
			MODEL	STK #	
			I/O System ERS		
			BY Fred Gross	DATE 09/20/82	
LTR	P.C. #	APPR	DATE	APPD	SHEET # 77 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

HEWLETT-PACKARD CO.

INPUT BYTE MATCH (NO MATCH)

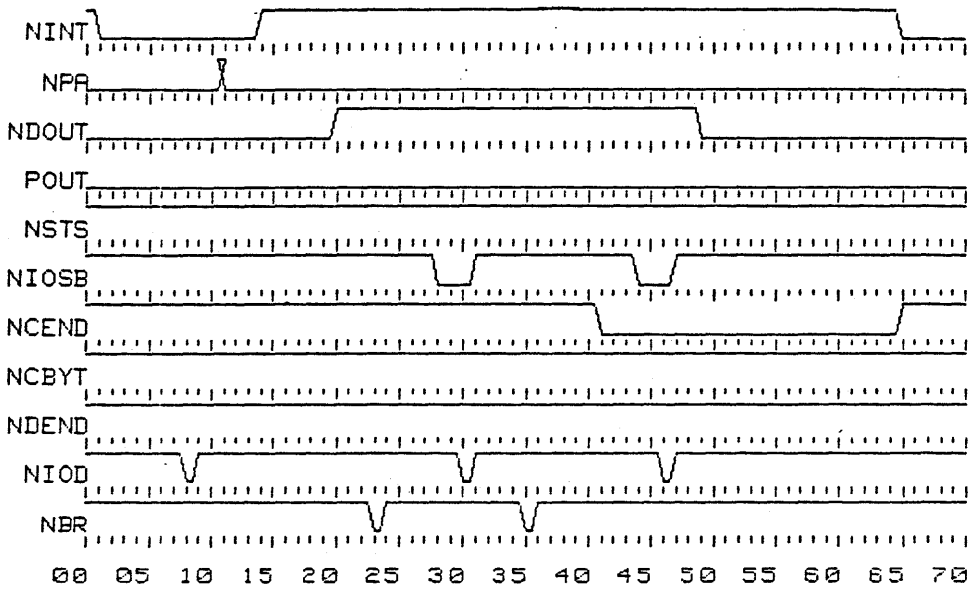


INPUT BYTE MATCH (MATCH)

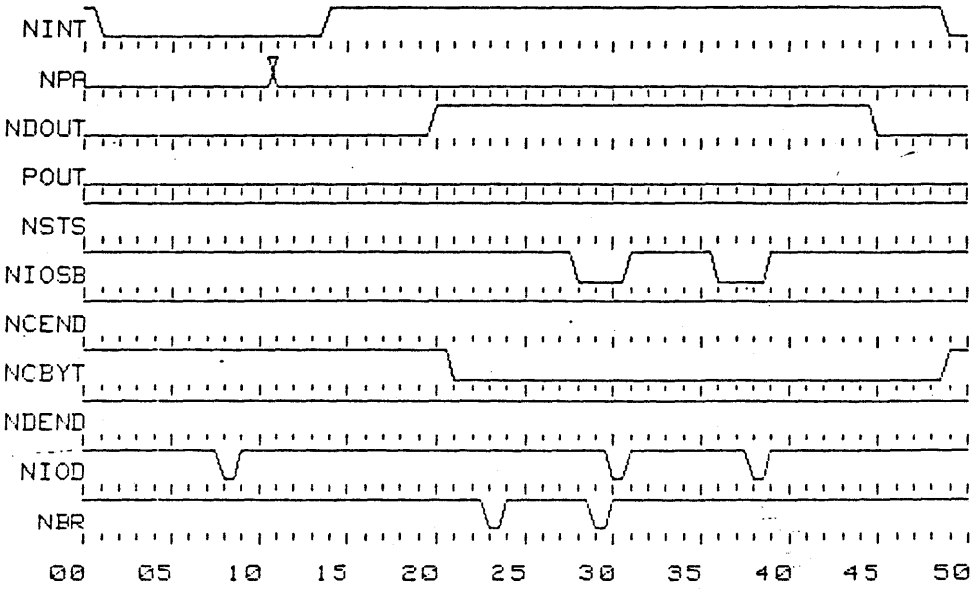


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
DT	P.C. #	APPR	DATE	APPD	SHEET # 78 OF 81
REVISIONS			SUPERSEDES		DWG # A-1FE1-3030-9

INPUT HALFWORD BURST - CHANNEL END

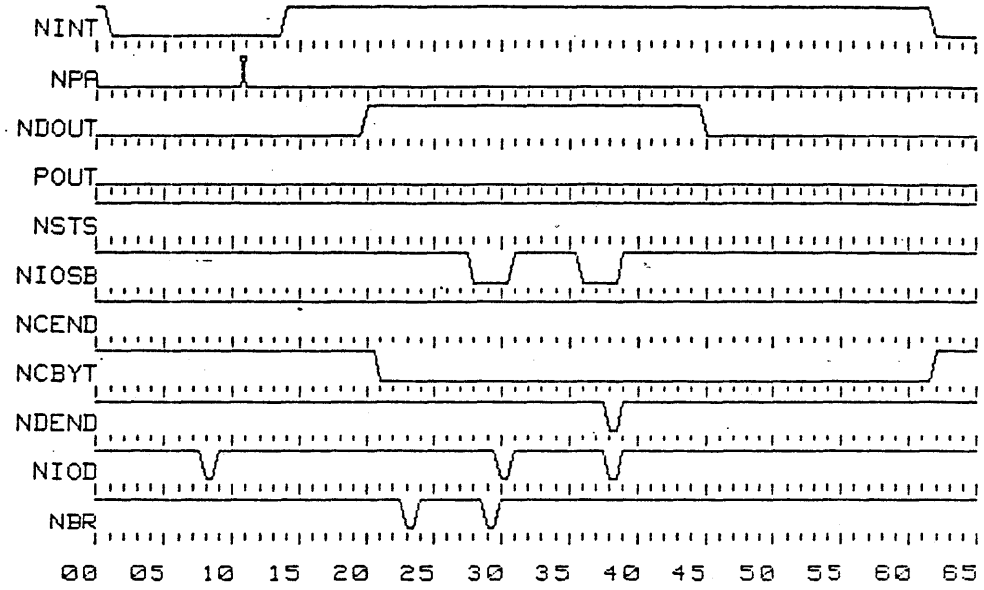


INPUT BYTE BURST

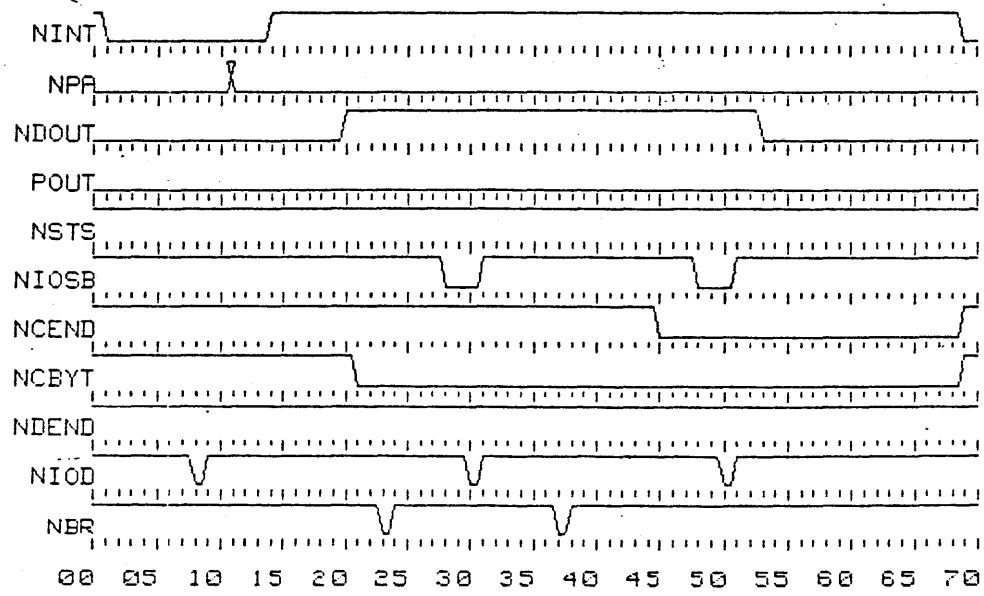


			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
P.C. #	APPR	DATE	APPD	SHEET # 79 OF 81
REVISIONS			SUPERSEDES	DWG # A-1FE1-3030-9

INPUT BYTE BURST - DEVICE END



INPUT BYTE BURST - CHANNEL END



			MODEL	STK #
			I/O System ERS	
			BY Fred Gross	DATE 09/20/82
LTR	P.C. #	APPR / DATE	APPD	SHEET # 80 OF 81
REVISIONS			SUPERSEDES	DWG # A-1FE1-3030-9