HEWLETT PACKARD

HP 1000 A-Series Computer

I/O Interfacing Guide

Measurement & Control Systems Divison
11000 Wolfe Road
Cupertino, CA 95014

Manual Part No. 02103-90055
E0292

Printed in U.S.A. February 1992
First Edition

NOTICE

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THE MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

RESTRICTED RIGHTS LEGEND

Use, duplication, or disclosure by the Government is subject to restrictions as set forth in subparagraph (c) (1) (ii) of the Rights in Technical Data and Computer Software clause at DFARs 252.227.7013

Safety and Regulatory Information

For your protection, this product has been tested to various national and international regulations and standards. The scope of this regulatory testing includes electrical/mechanical safety, radio frequency interference, ergonomics, acoustics, and hazardous materials. Where required, approvals obtained from third-party test agencies are shown on the product label. In addition, various regulatory bodies require some of the information under the following headings.

USA Radio Frequency Interference

The United States Federal Communications Commission (in 47CFR Subpart J, of Part 15) has specified that the following notice be brought to the attention of the users of this product:

Warning

This equipment generates, uses, and can radiate radio frequency energy, and if not installed and used in accordance with the instructions manual, may cause interference to radio communications. It has been tested for compliance with the limits for Class A computing devices pursuant to Subpart J of Part 15 of FCC Rules, which are designed to provide reasonable protection against such interference. Operation of this equipment in a residential area is likely to cause interference, in which case the user, at his own expense, will be required to take whatever measures may be required to correct the interference.

Japanese Radio Frequency Interference

The following notice is for users of this product in Japan:

この装置は，第一種情報装置(商工業地域において使用されるべき情報装置)で商工業地域での電波障害防止を目的とした情報処理装置等電波障害自主規制協議会（ＶＣＣＩ）基準に適合しております。

従って，住宅地域またはその隣接した地域で使用すると，ラジオ，テレビジョン受信機等に受信障害を与えることがあります。

取扱説明書に従って正しい取り扱いをして下さい。

Japanese Radio Frequency Interference

HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

German Radio Frequency Notice

This is to certify that this product meets the Radio Frequency Interference Requirements of Directive 1046/84. The German Bundespost has been notified that this equipment has been put into circulation and has been granted the right to check the product type for compliance with these requirements.

Funkentstörung Deutschland (German EMI Compliance)
Herstellerbescheinigung

Hiermit wird bescheinigt, daß dieses System in Übereinstimmung mit den Bestimmungen von Postverfügung 1046/84 funkentstört ist.

Der Deutschen Bundespost wurde das Inverkehrbringen dieses Systems angezeigt und die Berechtigung zur Überprüfung der Serie auf Einhaltung der Bestimmungen eingeräumt.

United Kingdom Telecom Statement (For the United Kingdom Only)

Warning

Interconnection of ports marked "UNITED KINGDOM TELECOM WARNING: Connect only apparatus complying with BS 6301 to these ports" with ports not so marked may produce hazardous conditions on the network and advice should be obtained from a competent engineer before such a connection is made.

Connect only apparatus complying with BS 6301 to the ports marked with the above warning.

Safety Considerations

GENERAL - This product and related documentation must be reviewed for familiarization with safety markings and instructions before operation.

SAFETY SYMBOLS

Instruction manual symbol: the product will be marked with this symbol when it is necessary for the user to refer to the instruction manual in order to protect the product against damage.

Indicates hazardous voltages.

Indicates earth (ground) terminal (sometimes used in manual to indicate circuit common connected to grounded chassis).

Explanation of Caution

Caution

The CAUTION sign denotes a hazard. It calls attention to an operating procedure, practice, or the like, which, if not correctly performed or adhered to, could result in damage to or destruction of part or all of the product. Do not proceed beyond a CAUTION sign until the indicated conditions are fully understood and met.

Explanation of Warning

Warning

The WARNING sign denotes a hazard. It calls attention to a procedure, practice, or the like, which, if not correctly performed or adhered to, could result in injury. Do not proceed beyond a WARNING sign until the indicated conditions are fully understood and met.

Printing History

The Printing History below identifies the edition of this manual and any updates that are included. Periodically, update packages are distributed which contain replacement pages to be merged into the manual, including an updated copy of this printing history page. Also, the update may contain write-in instructions.

Each reprinting of this manual will incorporate all past updates; however, no new information will be added. Thus, the reprinted copy will be identical in content to prior printings of the same edition with its user-inserted update information. New editions of this manual will contain new information, as well as all updates.

First Edition ..... Feb 1992 .....

Preface

This manual has been prepared as a guide in selecting the proper interfacing approach and as an aid in selecting or designing the appropriate hardware and software interfaces to the HP 1000 A-Series Computer I/O system. This manual complements the HP 1000 A400, A600/A600+, A700, A900, and A990 Computer Reference Manual. You should become thoroughly familiar with the contents of this manual and the applicable Computer Reference Manual before attempting to use the information presented in any part of this manual.

CHAPTER 1: INTRODUCTION — Presents an overview of the A-Series computer architecture, features, concepts, functional specifications, and an introduction to the ease of hardware and software interfacing.

CHAPTER 2: PROGRAMMING — Presents specific code segments that perform the functions of the I/O system. Included are any special features or considerations to simplify the task of the designer.

CHAPTER 3: HP INTERFACES — Provides a list of available general-purpose A-Series I/O interfacing cards and their manuals, and gives a brief description of the HP 12010A Breadboard interface.

CHAPTER 4: I/O MASTER — Describes the purpose, elements, and control of the I/O system and the I/O Master theory of operation. Also included are specifications and diagrams for the HP 12010A Breadboard; lists of signal names, mnemonics, and definitions; and signal timing diagrams and specifications.

Table of Contents

Chapter 1
Introduction

Chapter 2
Programming

Chapter 3
HP Interfaces

Chapter 4
I/O Master

List of Illustrations

Tables

1

Introduction

The HP 1000 A-Series Computers are LSI-based computers designed for users who require a low-priced computer that combines powerful computational capability with efficient Input/Output (I/O) processing. This chapter contains a general discussion of the HP 1000 A-Series Computer's operation and architecture, functional specifications, and interfacing requirements.

Computer Overview

The A-Series Computers comprise a central processor, memory, one or more input/output interface cards, and a power supply and backplane. The central processor (CPU) and I/O interfaces share control of the backplane and memory. The CPU and the I/O processor (IOP) chip have a master-slave relationship so that at any given time either the CPU or a single IOP is master of the bus and the other is a slave.

The backplane comprises a data bus, a memory address bus, and a control bus. The data bus holds data as it is being transferred between memory, the central processor and I/O interfaces. The address bus holds the address of the location to be referenced in memory. Either the central processor or the I/O chips may drive the address bus. The data bus is 16 bits wide, whereas the address bus is 20 bits wide; both buses are TTL compatible, and driven by tri-state logic.

A400 Computer

The A400 Computer is a single-board computer that consists of a CPU, memory controller, 512K bytes of parity memory, and an asynchronous serial I/O multiplexer, all on a single printed-circuit assembly. The A400 board plugs into a 6-slot, 14-slot, or 20-slot A-Series box with up to four parity memory arrays and as many standard A-Series I/O cards as the box will support. Most of the central processor unit (CPU) circuitry resides on a single CMOS chip.

All I/O instructions are executed by a custom input/output processor (IOP) chip that is part of the on-board I/O (OBIO) circuitry on the A400 board. The OBIO cable provides four 25-pin RS-232 serial I/O connections.

A600 and A600+ Computer

The A600/A600+ Computer design utilizes microprogrammed bit-slice chips in the CPU. The basic A600/A600+ computer (excluding I/O interface cards and expanded memory) consists of two printed circuit cards: a central processor card, and a memory controller card that includes either 512K or 1M bytes of dynamic RAM. The memory may be expanded up to 32M bytes by the addition of one to four memory array cards. (The A600 computer has been discontinued by HP and is discussed in this manual for reference only.)

The control logic is implemented as a microprogram contained in a PROM control store of 1024 56-bit microinstruction words. Decoding of 16-bit assembly language instructions is done by PROMs, which contain the entry point addresses in the control store associated with every defined opcode.

The memory accessing logic in response to control signals from the instruction decode logic arbitrates DMA versus processor contention for memory as well as generating the appropriate backplane signals for instruction fetching. The interrupt processing logic collects interrupt requests from various system and I/O level sources. An interrupt vector generator decides which interrupt is serviceable by passing a microcode entry point to the microprogram sequencer. This causes a microcode branch to the service routine for that interrupt.

The I/O accessing logic primarily participates in the IORQ/IOGO handshake protocol as used in all A-Series computers. It generates all of the necessary signals to send or receive data over the backplane or to freeze the micromachine if conditions do not permit the microcode to proceed further. One category of I/O instructions (STC/ CLC or STF/CLF) affects only the IOP, whereas the second category (SFS/C, LIA/B, MIA/B, or OTA/B) requires interaction with the CPU to modify the program counter or to access or modify the A/B-Register. The CPU during initial fetch of an I/O instruction determines if handshaking is required. An I/O state machine generates the necessary signal timing for communication over the backplane. For slave processing, I/O handshake protocol is used after the initial interrupt has been serviced.

A600+ computers are available with either of two memory systems. The standard memory system is based on a 512K-byte memory controller card, while the alternate memory system uses a 1M-byte memory controller. Main memory in either system may be expanded by the addition of up to four memory array cards, each having 512K, 1M, 2M, 4M, or 8M bytes of dynamic RAM with single-bit parity.

A700 Computer

The A700 Computer is a fully microprogrammable machine that uses bit-slice chips in the CPU and uses the A-Series distributed intelligence I/O system. The A700 computer consists of two processor cards, a memory controller card, from one to four memory array cards, and a floating point processor card. Data and addresses are exchanged between the computer cards over frontplane buses and the backplane links the computer cards, the I/O system cards, and power supply. The control store resides on the lower processor card (base instruction set), on the floating point processor card (instruction set enhancements), and optional writable control store or PROM control store cards. (The A700 computer has been discontinued by HP and is discussed in this manual for reference only.)

In operation, assembly language instructions are translated through a look-up table into addresses in the 16k 32-bit word control store. The microinstructions from the control store are translated into control signals by microinstruction decode logic. The base instruction set (2k 32-bit words) of the control store resides on the lower processor board and the upper processor card contains the memory and I/O access logic.

The CPU generates low select-code system level interrupts such as memory protect, time base generation, and unimplemented instruction interrupt; logic is provided for detecting other system level functions such as power fail/ auto restart and parity error interrupt.

The A700 memory system consists of a memory controller card and up to four memory array cards.

A900 Computer

The A900 computer features a high-speed microprogrammed control processor that resides on four cards called the sequencer, data path, cache control, and memory controller. The high speed is achieved through "pipelining" of the next instruction sequence and a cache memory to speed up memory accesses. The A900 uses the same distributed intelligence concept as the other A-Series computers that separates the processing of I/O instructions from that of other instructions.

Data and addresses are exchanged between the memory controller and memory array card or cards over a frontplane. If a control store card is installed, there is a separate frontplane going between the control store and sequencer cards. The backplane links the computer card, the I/O system cards, and power supply. The internal control store resides on the sequencer card and an optional control store card includes 4k words of writable control store and 2k words PROM control store. Arithmetic and floating point functions are contained on the data path card. Each microword is 48 bits wide.

The computer is always under microprogram control and executing microinstructions when power is applied. Assembly language instructions are translated through a look-up table with pointers to microroutines in control store that implement the instructions. There are 6k words of ROM storing the A900 base set.

Most of the input/output logic resides on the cache card and the macrointerrupt logic is on the memory controller card. The CPU generates low select-code interrupts such as memory protect, time base generation, multi-bit error, and unimplemented instruction. Logic is provided for detecting other system functions such as power fail/auto restart and parity error interrupt.

The A900 memory system consists of a memory controller and up to eight memory array cards.

A990 Computer

The A990 Computer is a single-board processor and is the fastest HP 1000 Computer. Pipeline technology with two-level cache memory, boosted even more by a fast built-in hardware floating point processor having both scientific and vector instruction sets in firmware, provides unmatched computational speed. The A990 design is based on the A900 Computer but uses proprietary

application-specific integrated chips (ASICs) to greatly improve overall performance. With 3/4-Mbyte, 3-Mbyte, and 8-Mbyte Error Correcting Code (ECC) memory cards available, and card cage space for up to four memory cards, the A990 memory can be a total of up to 32 Mbytes.

A-Series Backplane

The backplane provides the link between the central processor, memory, interface cards, and power supply. It is a printed circuit card on which the traces distribute the power, ground, and interconnecting signals between all the cards in an A-Series Computer. The backplane is the hardware communications link between all cards in the system. Communication with a device interface is handled by the I/O Master. This I/O Master circuitry located on every A-Series interface card serves to standardize the I/O interface to the backplane. This frees the I/O designer from the details of interfacing to the backplane and allows concentration on the device specific interface design. -

I/O System

The A-Series I/O system allows data to be transferred between a computer and external devices either through the central processor A- and B-Registers (programmed I/O) or directly to memory (direct memory access). The external device itself may be physically located on the interface, such as the PROM Storage Module. The I/O interface logic has been optimized for direct memory access (DMA) data transfers. Once this DMA logic, which is included on every interface, has been initialized, no further programming is involved and the transfer to or from memory is automatically handled by the interface. This transfer occurs in two distinct steps as follows:

a. Between the external device and its I/O interface in the computer.

b. Between the I/O interface and memory via the backplane data bus.

An output transfer uses the same process except, in reverse order.

Data may be transferred under program control without using the DMA capability. This type of transfer allows the computer to manipulate the data during the transfer process. A programmed I/O input transfer is a three step process as follows:

a. Between the external device and its I/O interface.

b. Between the I/O interface and the A- or B-Register via the data bus and the central processor.

c. Between the A- or B-Register and memory via the processor and the data bus.

Note that during a DMA transfer the central processor is bypassed. Since a DMA transfer eliminates programmed steps that load data into the A- and B-Registers and store the data directly into memory, the DMA transfer time is very short. Thus the DMA capability is especially useful with high-speed devices that may transfer data at rates up to 3.9 megabytes per second

(A700), 3.7 megabytes per second (A900), 4.0 megabytes per second (A990), or 4.3 megabytes per second (A400/A600+). DMA transfers also eliminate time consuming processor interrupts that are associated with each programmed I/O data transfer. A self-configuration feature of the I/O Master (to be discussed later) further reduces the processor interrupts by allowing the next DMA transfer to fetch its set of control words directly from memory, to reconfigure and enable itself automatically.

I/O Addressing

As shown in Figure 1-1, an external device is connected by cable directly to an I/O interface located in the computer mainframe. The I/O interface, in turn, plugs into one of the input/output slots in the backplane. Note, however, that the select code (I/O address) of each I/O interface is independent of its priority. The computer communicates with a specific device on the basis of its select code, which is set by switches on the I/O interface.

Figure 1-1 shows an I/O interface inserted in the I/O slot having the highest priority (in the A900 this I/O card may be replaced with the control store card). If it is decided that the associated device should have lower priority, its I/O interface and cable may simply be exchanged with those occupying some other I/O slot. This will change the priority but not the I/O address (select code). Due to priority chaining, there can be no vacant slots from the uppermost memory card slot to the lowest priority slot used. Only select codes 20 through 77 (octal) are available for I/O interfaces; the lower select codes (00 through 17) are reserved.

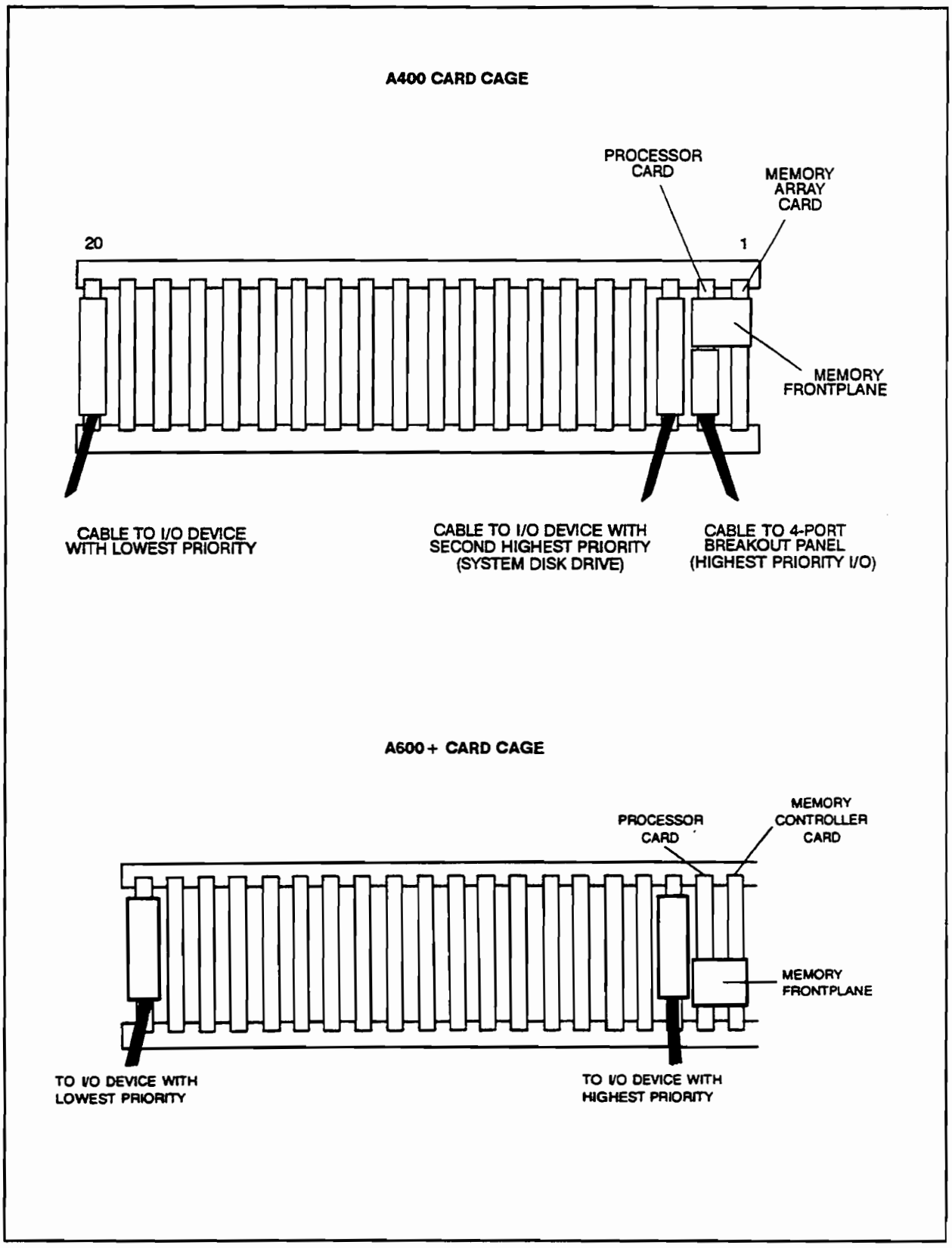


Figure 1-1. I/O Priority Assignments (Sheet 1 of 4)

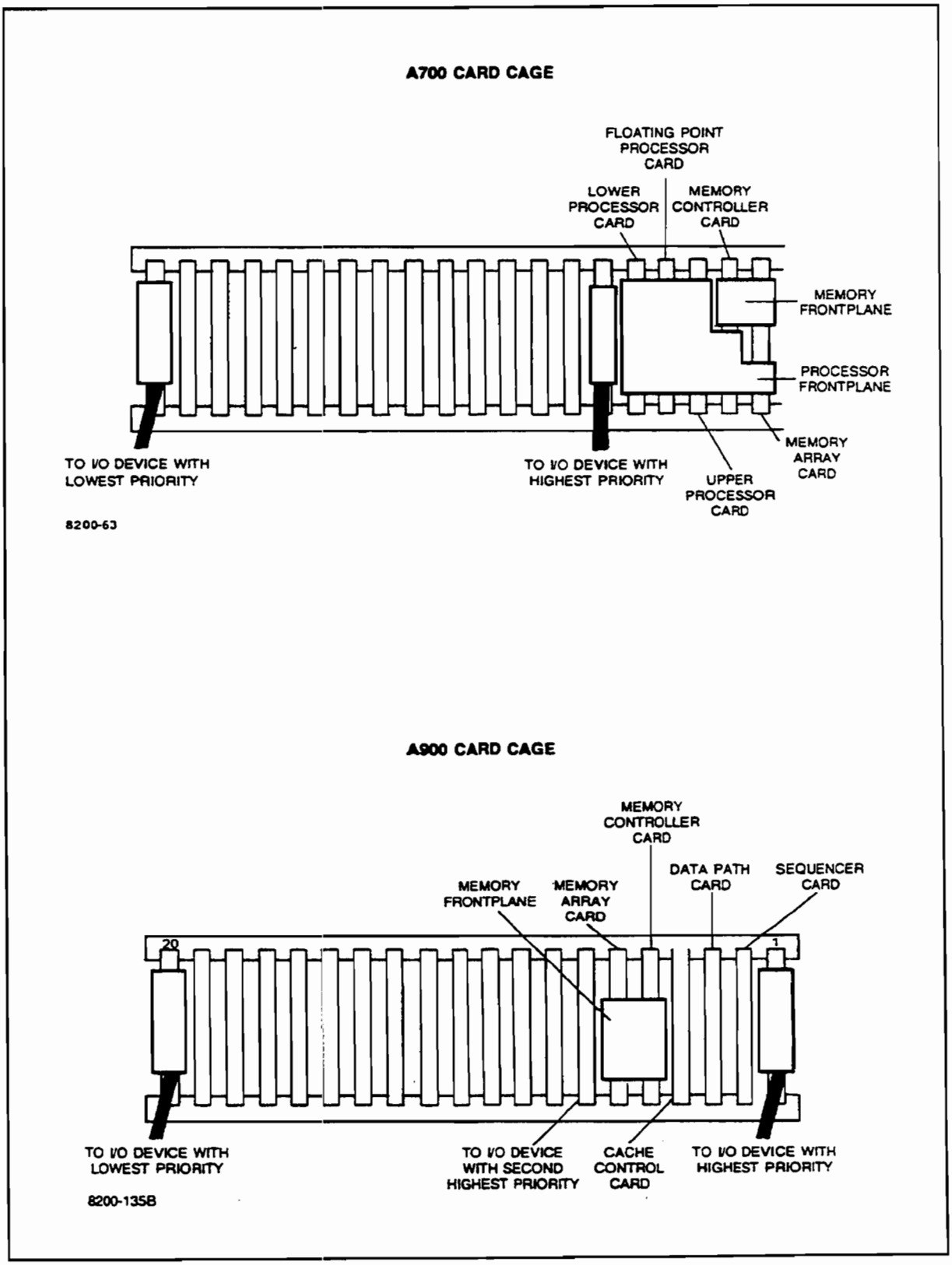


Figure 1-1. I/O Priority Assignments (Sheet 2 of 4)

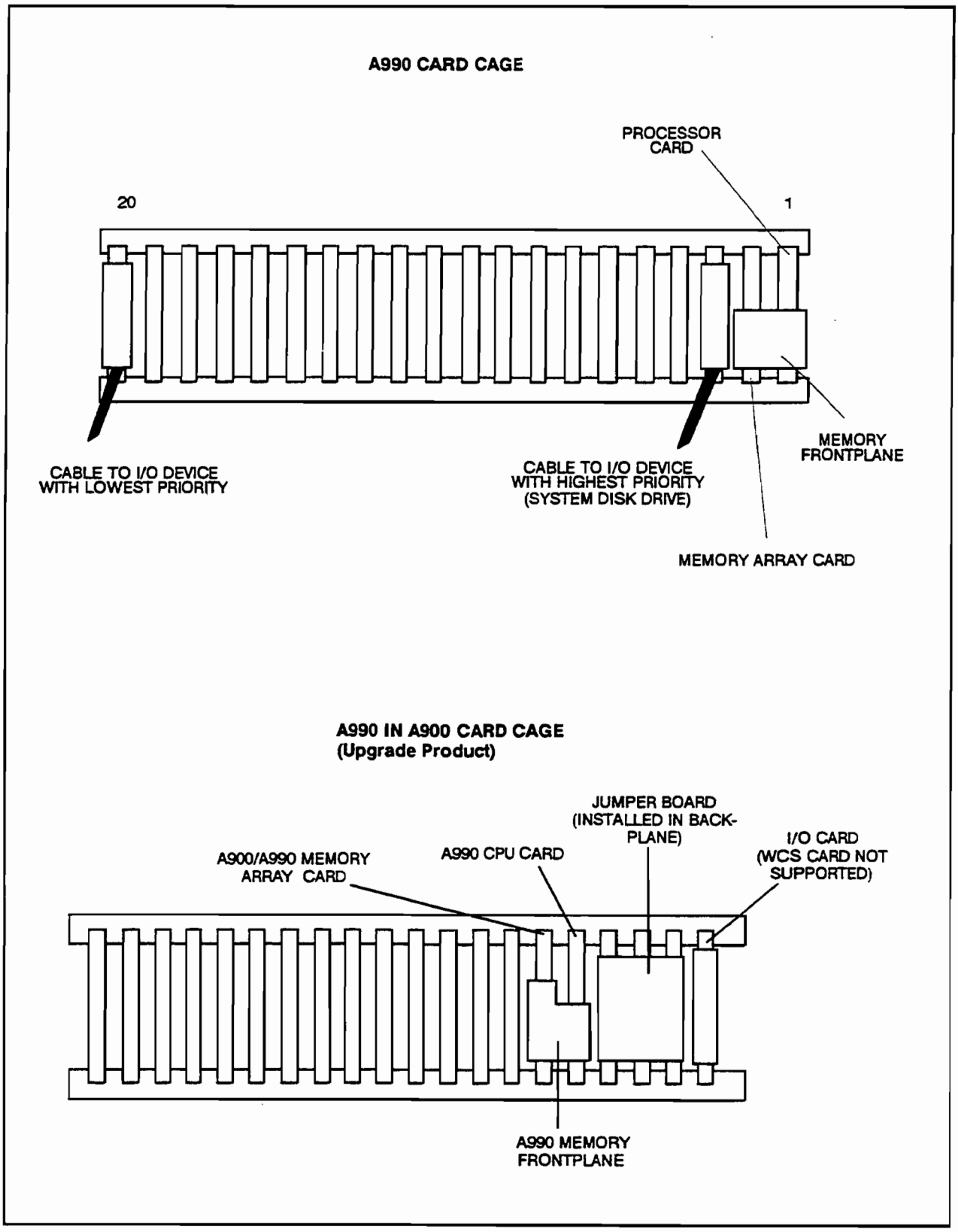
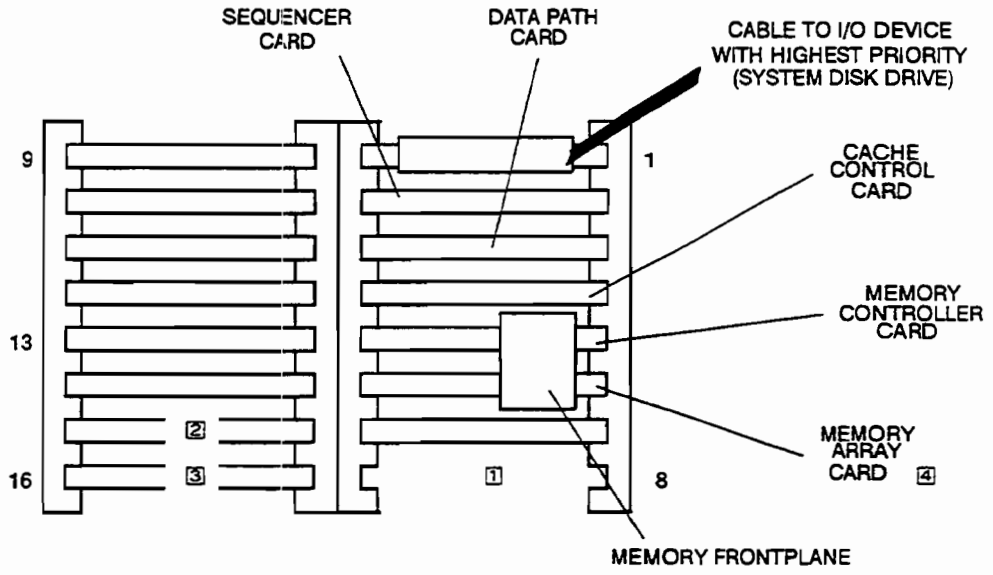


Figure 1-1. I/O Priority Assignments (Sheet 3 of 4)

MICRO/1000 CARD CAGE*



Notes:

1. Slot 8 is reserved for the optional HP 12159A 25 kHz Power Module. If not installed, this slot must remain unoccupied.

2. Slot 15 is the lowest priority I/O card if optional HP 12154A Battery Backup Card is not installed. If the battery backup card is installed, see note 3.

3. Slot 16 is reserved for the HP 12154A Battery Backup Card which, when used, extends into the space occupied by slots 14 and 15. This makes slot 13 the lowest priority I/O card slot. If the battery backup card is not installed, a voltage jumper card (part no. 02430-60009) must be installed in slot 16.

4. One or two memory array cards are supported.

* The A900 configuration is shown. The other models are the same except for the card and frontplane configurations as shown in the previous diagrams.

Figure 1-1. I/O Priority Assignments (Sheet 4 of 4)

I/O Priority

The plug-in card slots of the A-Series computers are numbered consecutively from top to bottom (starting in the right side if there are two columns) for those configurations with horizontal slots or from right to left with vertical slots. Generally, memory occupies the highest priority slots, then the central processor and remaining slots are available for I/O interfaces. The slot immediately following the Processor card will have the highest I/O interrupt priority except for the A900 processor where the first I/O slot is immediately above the sequencer card, unless it is occupied by a control store card.

When an input/output device is ready to be serviced, it causes its I/O interface to request an interrupt so that the computer will interrupt the current program and service the device. Since many device interfaces may request service at random times, it is necessary to establish an orderly sequence for granting interrupts. Also, it is desirable that high-speed devices should not have to wait for low-speed device transfers. Both of these requirements are met by a series-linked priority structure, illustrated by Figure 1-2. The bold line, representing a priority enabling signal, is routed in series through each card. The card cannot interrupt unless this enabling signal is present at its input.

Each device (or other interrupt function) can break the enabling line when it requests an interrupt, by pulling its signal line low. If two devices simultaneously request an interrupt, the device with the highest priority will be the first one that can interrupt because it has broken the enable line for the lower-priority device. The other device cannot begin its service routine until the first device is finished. However, a still higher-priority device (one interfaced through a lower-numbered slot) may interrupt the service routine of the first device. Figure 1-3 illustrates a hypothetical case in which several devices request service by interrupting a CPU program. Both simultaneous and time-separated interrupt requests are considered.

Assume that the computer is executing a program when an interrupt from the I/O interface located in slot 5 occurs (at reference time t1), and that the card in slot 5 is assigned select code 22. With the I/O card supplying the select code to the address bus, a JSB instruction in memory location 22 causes a program jump to the service routine for the interface in slot 5 (select code 22). The JSB instruction automatically saves the return address (in a location which the programmer must reserve in the service routine) for a later return to the CPU program. The program counter is not affected by the interrupt and the subsequent execution of the JSB at location 22, so effectively the JSB is "inserted" into the program at whatever point the interrupt occurs. (The JSB is not physically inserted, so that later passes through the same section of code would not JSB to the interrupt routine unless another interrupt occurred.)

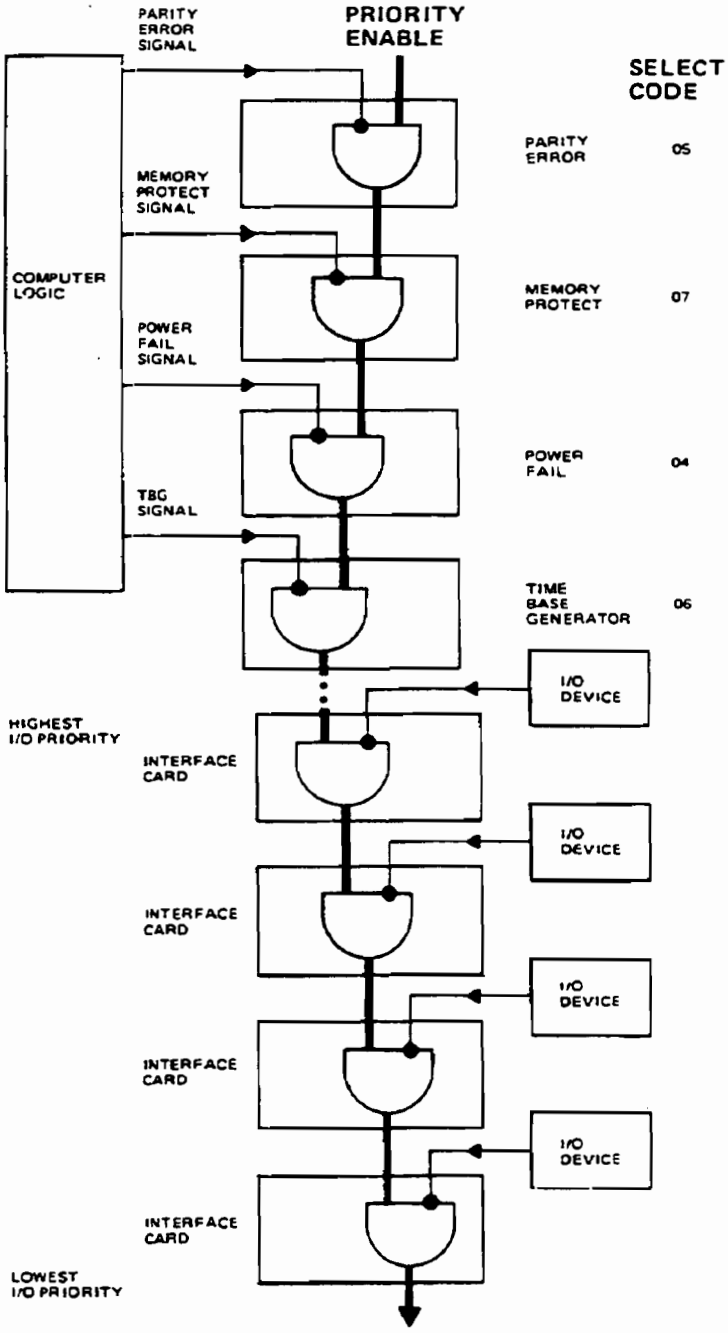
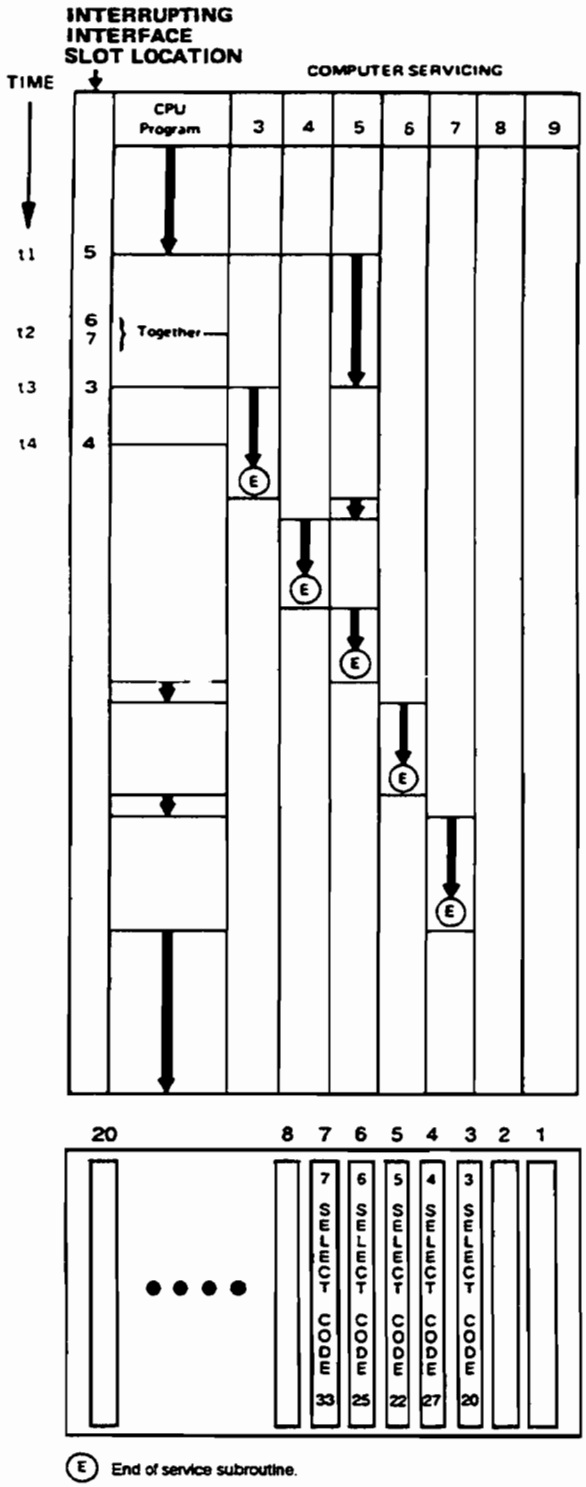


Figure 1-2. Priority Linkage (simplified)



Note: The A600+ is represented. The other CPUs are the same except shifted in slot location by their different processor configurations.

Figure 1-3. Interrupt Sequences

The routine for select code 22 is still in progress when several other devices request service (set flag). First, interfaces in slots 6 and 7 (select codes 25 and 33) request simultaneously at time t2; however, as neither one has priority over the interface in slot 5, their flags are ignored and the routine for select code 22 continues executing. But at t3, a higher priority (select code 20) requests service. This request interrupts the select-code 22 routine and causes the select-code 20 routine to begin. The JSB instruction saves the return address for return to the select-code 22 routine.

During the transfer to the slot 3 interface's service routine, the flag on the interface in slot 4 (select code 27) is set (t4). Since it has lower priority than the interface in slot 3, the slot 4 interface must wait until the end of the select-code 20 routine. Because the select-code 20 routine when it ends contains a return address to the select-code 22 routine, program control temporarily returns to the select-code 22 routine, even though the waiting interface in slot 4 has higher priority. The JMP,I instruction used for the return inhibits all interrupts until after the next instruction is executed. At the end of this short interval, the select-code 27 interrupt request is granted.

When the select-code 27 routine is finished, control is returned to the select-code 22 routine, which at last has sufficient priority to complete its routine. Since the select-code 22 routine has been saving a return address to the main program, control returns to this program.

The two waiting interrupt requests from select codes 25 and 33 can now be considered. The interface in slot 6 has the higher priority and goes first. At the end of the select-code 25 routine, control is temporarily returned to the CPU program. Then the select-code 33 interface interrupts and completes its routine. Finally, control is returned to the CPU program, which resumes processing.

I/O Processing

In the A-Series, most of the I/O group instructions are not executed by the CPU chip or micromachine. Instead, they are executed by the IOP chip on the interface to which the instruction pertains. Instructions such as LIA, OTB, MIA, or that require the executor to read or modify some register in the CPU are executed using a special I/O access protocol. When an I/O interface determines that it should execute the instruction just fetched, it also decides if the execution will require interaction with the CPU. If the CPU is needed, the interface will request an I/O handshake over the backplane to the central processor. In the meantime, the central processor has determined that it is not responsible for the execution of the instruction, and has prepared itself for a possible I/O handshake request.

The CPU acknowledges the I/O handshake request after any active and pending DMA memory requests are complete. The interface that requested the handshake responds by driving a command word onto the data bus at the completion of the handshake.

For some instructions, such as skip, passing a command to the CPU is all that must be done. For instructions that require transfer of data, such as OTA or LIA, two handshakes are required. In the first handshake, the control word passed to the CPU tells it what to do for the second handshake. The control word will select the A- or B-Register and set the direction of the data transfer. The CPU will either write to or read from the data bus during the second handshake, while the I/O interface performs the opposite operation.

According to their execution requirements, I/O instructions may be broken down into three groups as follows:

a. Data Transfer I/O Instructions—OTA/B, LIA/B, MIA/B

   This group requires a double handshake. In the first handshake, a control word is transferred from the I/O interface to the central processor. In the second handshake, the data is transferred either into or out of the A- or B-Register, according to which of the six instructions is being executed. I/O transfers over the backplane to the central processor have lower priority than DMA transfers, and can be preempted.

b. Status Sensing I/O Instructions—SFS, SFC

   This group requires, at most, a single handshake during which a control word (signalling the CPU to increment P) is transferred from the I/O interface to the central processor. If no skip is required, no handshake occurs.

c. Status Altering Instructions—STC, CLC, STF, CLF

   This group affects only the control or flag flip-flops on the I/O processor, and requires no interaction with the central processor.

Slave Mode Transfers

An I/O interface may force the central processor to enter the slave or Virtual Control Panel (VCP) mode. When the central processor is operating in slave mode, it is under control of the slave mode requestor, and thus the device connected to the requestor becomes a virtual control panel. As such, this device can initiate I/O transfers as described above in step a.

Interrupt Processing

The A-Series interrupt system allows an external, asynchronous event to obtain the attention of the central processor without the processor having to specifically check the status of the I/O interfaces periodically. On acknowledging an interrupt, control may pass directly to a routine that services the interrupt. This is especially useful for time-critical processes that must receive a quick response. The interrupt system allows the operating system to suspend the currently executing program, respond to the time-critical event, then resume processing the suspended program.

An interrupt request occurs when an interface control is set and the flag is set either by the device itself or by the execution of an STF instruction. This causes the interface to assert an interrupt-requesting signal on the backplane. An interrupt acknowledgement, from the central processor, is triggered by an interrupt request from any of the I/O interfaces. When the CPU is ready to fetch the next instruction, and if the interrupt system is enabled and interrupts are not temporarily being held off, then the central processor will assert an interrupt acknowledge signal.

Since interrupt servicing is accomplished with the aid of a memory cycle, the interrupt handshake timing is similar to that of a memory cycle. After the interrupt is acknowledged and memory is ready, the interface drives its select code on the address bus. The CPU then executes the instruction held in the memory location corresponding to the interrupting interface's select code. This instruction will normally be a JSB,I to the interrupt service routine. All such service routines

normally end with an XJMP indirect instruction to the beginning of the service routine, which will return the computer to the program that was executing when the interrupt occurred. For example, if the interrupting interface has select code 25, the CPU executes the instruction in memory location 25, which should hold a JSB,I to the service routine. When the interface requested an interrupt, it also issued a signal to disable all lower-priority interfaces from requesting interrupt service. If a higher-priority interface preempts the request, the lower priority requesting card will be disabled. The lower-priority interface should maintain its request until the disabling signal is terminated and the lower-priority interface can be serviced.

Note

In an A-Series computer, only JMP(,I) or JSB(,I) may be used in an interrupt trap cell.

The interrupt system flag (FLG0) is set and cleared under program control. It permits a programmer to specify which parts of a program may be interrupted for I/O and which may not.

Interface Elements

The I/O interface provides the communication link between the computer and one or more external devices. The I/O interface includes several basic elements, some of which either the computer or the device can control in order to effect the necessary communication. These basic elements are the Global Register, control flip-flop, flag flip-flop, data buffer register, card control register, and DMA control registers. DMA control registers are discussed in Chapter 2 under DMA.

Global Register

The A-Series computer utilizes a design feature called the "Global Register" for addressing interfaces. Every I/O interface card (serial interface, parallel interface, etc.) in the HP 1000 A-Series Computer System contains a Global Register. The Global Register is located in the IOP chip and is a six-bit register which contains a select code. The select code of a particular I/O interface is set by switches on that interface.

All Global Registers on all interface cards are controlled by the CPU, thus all Global Registers contain the same select code. (It is useful, therefore, to think of all the Global Registers as being one "global" register.) When the Global Register is enabled, any I/O instruction fetched by the CPU automatically applies only to the card whose select code matches the value stored into the Global Register. For example, if the Global Register is enabled and contains a value of 22, then the current I/O instruction is decoded and executed only by the interface with a select code of 22.

Using the Global Register to address I/O interfaces allows the six least significant bits of the I/O instruction to be used to address specific registers on the I/O interface cards. (These six bits are not needed for identification of the I/O interface to which the instruction applies since that is already taken care of by the Global Register.) For example, with the Global Register previously set to a value of 22, an OTA instruction with a value of 31 stored in the six least significant bits of the

instruction field would cause the contents of the A-Register to be output to register 31 of the I/O interface card having a select code of 22.

Control Bits

The control bits on an I/O interface are used to initiate I/O functions. In addition, a control bit must be set to allow the corresponding flag bit to cause an interrupt. There are three control bits associated with each I/O select code: control 20, 21, and 30. When control 30 is set, it generates an action command, allowing one word or character to be transferred to or from the data buffer. When control 20 is set, it turns on DMA self-configuration (refer to Chapter 2 under DMA). The setting of control 21 starts a DMA transfer.

Flag Bits

The flag bits, when set, are used primarily to interrupt or to signal task completion. Flag 30, the only flag accessible without using the Global Register, signals either one data element has been transferred or that an interrupting condition has been detected. There are five other flags. Flag 21 signals DMA transfer complete; flag 20 signals DMA self-configuring transfer complete; and flag 22 signals that a parity error occurred during a DMA memory access. The device cannot clear the flag bits. If the corresponding control bit is set and the interrupt system is enabled, then setting the flag bit will cause an interrupt to the location corresponding to the I/O interface's select code. Flag 23 is the logical OR of flags 20, 21, and 22. It is used to distinguish between DMA and I/O interrupts. If flag 23 is not set, the interrupt is not DMA related. Flag 24 may be used to determine whether DMA is enabled (set) or disabled (clear).

Data Buffer Register

The data buffer register (designated Register 30 by HP convention) is used for the intermediate storage of data during an I/O transfer. Typically, the data capacity is 16 bits. The data register is not part of the I/O Master, it must be designed into the user interface logic.

Card Control Register

The card control register (designated Register 31 by HP convention) enables a general purpose I/O interface to be configured for compatibility with a specific I/O device or to be programmed for particular modes of operation. The control register must be programmatically set up (GR enabled, OTA/B 31) for a particular application. The control register is not part of the I/O Master; it must be designed into the user interface logic.

Functional Specifications

Table 1-1 lists the functional specifications for the A-Series computers.

Table 1-1. Functional Specifications for A-Series Computers

ARCHITECTURE

Type:	Distributed intelligence with separate CPU and I/O Processors communicating with each other and with memory via a single bus.
Implementation:	Microcoded with bipolar LSI and MSI hardware.
Data Path Width:	16 bits.
Bus Structure:	Single backplane bus for memory, processor, I/O.
Backplane Bus Speed:	A400: Up to 4.3M bytes/sec. A600/A600+: Up to 4.3M bytes/sec. A700: Up to 3.9M bytes/sec. A900: Up to 3.7M bytes/sec on inpur; up to 2.5M bytes on output. A990: Up to 4.0M bytes/sec on input; up to 2.7M bytes on output.
Interrupt System:	Vectored priority interrupt structure with the following assignments in priority.

Select Code/ Memory Location	Function
00005	Memory parity interrupt (A900/A990: multibit error)
00010	Unimplemented instruction interrupt
00007	Memory protect interrupt
00004	Power fail interrupt
00006	Time base generator interrupt
00011–00016	Reserved
00020–00077	I/O device interrupts

CENTRAL PROCESSOR

Accumulators:	2 (A and B), 16 bits each. Implicitly addressable, also explicitly as memory locations.
Index Registers:	2 (X and Y), 16 bits each.
Supplementary Registers:	2 (overflow and extend), one bit each.
Power Fail Provisions:	When primary line power falls below a predetermined level while the computer is running, a power fail warning signal from the power supply causes an interrupt to memory location 00004. Memory location 00004 is intended to contain a jump-to-subroutine (JSB) instruction to a user-written power fail subroutine. A minimum of 5 milliseconds is available to execute the power fail subroutine.
Memory Protect:	CPU memory protect logic (when enabled): 1. Protects memory on a page-by-page basis against alteration or entry by programmed instructions, except those involving the A- and B-Registers.

Table 1-1. Functional Specifications for A-Series Computers (Continued)

	2. Prohibits execution of I/O instructions except those referencing select code 01 (the CPU status register and overflow register). This limits control of I/O operations to interrupt control only, which can be used to give exclusive control of the I/O system to an executive program. 3. In response to a memory protect violation, interrupts the computer and saves the address of the violating instruction in the memory protect violation register, from which it can be made accessible in the A- or B-Register by a single assembly language instruction.
Time Base Generator Interrupt:	A time base generator interrupt is provided for maintaining a real-time clock. The interrupt request is made when the CPU signals, at 10-millisecond intervals, that its clock is ready to roll over. Timing accuracy of the TBG is plus or minus 2.16 seconds per day. The A990 also allows setting the TBG to a user-selected frequency.
Unimplemented Instruction Interrupt:	An unimplemented instruction interrupt is requested when the CPU chip signals that the last instruction fetched was not recognized by it. This interrupt provides a straightforward entry to software routines for the execution of instruction codes not recognized by the A-Series system hardware.
MEMORY	
Memory Structure:	64 pages minimum of 2048 bytes per page, with direct access to cur rent page or base page (page 00), and indirect or mapped access to all other pages.
Memory Cycle Time:	A400: 454 nanoseconds. A600/A600+: 454 nanoseconds. A700: 500 nanoseconds. A900: 133 nanoseconds (cache memory) A990: 100 nanoseconds (cache memory)
I/O MASTER	
Purpose:	To maintain the high performance of the A-Series I/O structure, HP uses the I/O Master as the standard interface circuitry to the back plane. It includes the I/O processor (IOP) chip, which executes I/O instructions, and other circuitry to make high-speed transfers possible. Every HP 1000 A-Series I/O interface card including the HP 12010A Breadboard interface, has the I/O Master circuitry.
Determination of I/O Address:	The I/O address select code is set for each interface by select code switches on the interface card and is therefore independent of the card position along the backplane bus. This select code address is loaded into a register in the IOP on power up and is compared with any I/O address instruction which is executed.
I/O Addressing:	An I/O interface is pre-addressed by having its select code loaded into the Global Register (GR) of the IOP chip, with a single instruction. This leaves the six select code bits of I/O instructions available for addressing registers or other functions on the interface.

Table 1-1. Functional Specifications for A-Series Computers (Continued)

I/O Instruction Execution Times:

Instruction	Execute Time (μsec)* A400	A600+	A700	A900	A990
STC,CLC,STF,CLF	2.66	2.95	3.50	**	***
LIA/B	5.31	5.90	6.00	3.07	5.30
MIA/B	5.31	5.90	6.00	3.07	5.30
OTA/B	4.50	4.99	5.25	3.33	5.30
SFC,SFS without skip	2.66	2.95	3.50	1.33	3.80
SFC,SFS with skip	3.67	4.09	5.25	2.40	3.80

* These figures assume no DMA intervention; DMA intervention may increase execution times.
** STC, CLC = 2.667; CLF, STF = 1.2
*** STC, CLC = 3.6; CLF, STF = 1.5

I/O Device Interrupt Priority:

Depends on I/O interface position along the card cage bus. The interface closest to the central processor card has the highest priority, with the other interfaces in card slots farther from the central processor having successively lower priorities.

I/O Interrupt Procedure:

One or more I/O interfaces requests an interrupt. When the central processor acknowledges the interrupt request, it executes the instruction in a memory location that corresponds to the select code of the highest-priority interrupting interface. The select code is placed on the address bus by the interface.

Interrupt Masking:

The I/O Master logic includes an interrupt mask register which provides for selective inhibition of interrupts from specific interfaces under program control. This register can be programmed to temporarily cut off undesirable interrupts from any combination of interfaces when they could interfere with crucial transfers.

Interrupt Latency:

A400: 4.2 to 36 microseconds; 4.6 microseconds typical. (Interrupts cannot be serviced until a DMA cycle or an instruction in progress has completed execution.) The worst-case latency of 36 microseconds is based upon time to complete the longest uninterruptible standard instruction.

A600/A600+: 4.7 to 40 microseconds; 5.1 microseconds typical. (Interrupts cannot be serviced until a DMA cycle or an instruction in progress has completed execution.) The worst-case latency of 40 microseconds is based upon time to complete loading or storing of a map (LMAP, SMAP), the longest standard instruction.

A700: 8.00 to 29.75 microseconds; 10.00 microseconds typical. (Interrupts cannot be serviced until a DMA cycle or an instruction in progress has completed execution.) The worst-case latency of 29.75 microseconds is based upon time to complete a floating point divide (FDV), the longest uninterruptible standard instruction.

A900: 3.7 to 13 microseconds; 4 microseconds typical. (Interrupts cannot be serviced until a DMA cycle or instruction in progress has completed execution.) The worst case latency is 13 microseconds based upon time to complete the longest uninterruptible standard instruction.

Table 1-1. Functional Specifications for A-Series Computers (Continued)

	A990: 3 to 15 microseconds; 3 microseconds typical. (Interrupts cannot be serviced until a DMA cycle or instruction in progress has completed execution.) The worst case latency is 15 microseconds based upon time to complete the longest uninterruptible standard instruction. Note: Interrupt latency is defined as the time interval between the generation of an Interrupt Request (INTRQ) signal by an I/O device and entry into the service routine.
Direct Memory Access (DMA):	The IOP chip supports DMA capability on each I/O interface, which reduces the number of interrupts from one per data item (byte or word) to one per complete DMA block transfer, greatly reducing over head and increasing throughput.
Self-Configured Chained DMA:	The IOP chip also supports a self-configuring mode of operation. In this mode, instead of interrupting after a block transfer, the IOP chip fetches a new set of control words for the next transfer, reconfigures itself and initiates another block transfer. This process continues for as long as additional control words are available.
Data Packing Under DMA:	When byte mode is specified in the DMA control word (Reg. 21) instructions, the IOP chip automatically packs or unpacks bytes.
Maximum Achievable DMA Rate:	A400: 4.27 megabytes/second. A600/A600+: 4.27 megabytes/second. A700: 4.0 megabytes/second. A900: 3.7 megabytes/second on input; 2.5 megabytes/second on output A990: 4.0 megabytes/second on input; 2.6 megabytes/second on output
Self-Configured Mode DMA Latency:	Maximum time between successive transfers of a chained transfer series (if not delayed by a higher priority DMA transfer): A400: 4.5 to 7.1 microseconds. A600/A600+: 4.5 to 7.1 microseconds. A700: 5.0 to 7.7 microseconds. A900: 5.2 to 8.1 microseconds. A990: 5.1 to 8.0 microseconds.
I/O MASTER SIGNALS AND TIMING: Refer to Chapter 4.	

Introduction to Interfacing

Interfacing a peripheral device to an HP 1000 A-Series Computer involves both hardware and software as shown in Figure 1-4. The software interface consists of an interface driver and device driver working with the RTE-A software as shown in the figure. The software interface can be accomplished by using existing computer I/O software, revising or writing new device drivers, or by writing new interface and device drivers. The hardware interface is accomplished by inserting one or more interface printed-circuit assemblies (PCAs) into easily accessible input/output (I/O) slots in the computer backplane and connecting to the peripheral device. The computer provides a unique I/O address (select code) and service priority interrupt for every I/O slot used. The hardware interface can be either an existing I/O card available from Hewlett-Packard or the user's own interface.

Interface/Device Drivers

A driver may be separated into two parts, the device driver and the interface driver, or remain as a single interface driver. The interface drivers manage the flag and control signals of the interface card. The device drivers would be designed to the particular characteristics of the devices they address. This driver organization provides high flexibility, and easy low cost development when a new device is added, because only a device driver may need to be designed. The separated driver concept proves most useful when there are several possible device types cabled to the same interface, such as HP-IB devices.

This technique enhances the ease by which new devices may be connected to interface cards. Using an existing interface driver, only a new device driver, or a modified existing driver, is needed to handle the characteristics of that new device. Another advantage is the ease of changing some functions of the device by accessing the device driver only. This could be done even if the interface card were busy at the time.

The device driver formats the output buffers or interprets the input buffers according to the characteristics of the device. If the device driver is not used then individual programs must perform these functions.

It may be helpful, when writing or debugging a driver, to refer to the *RTE-A Driver Reference Manual*, part number 92077-90011.

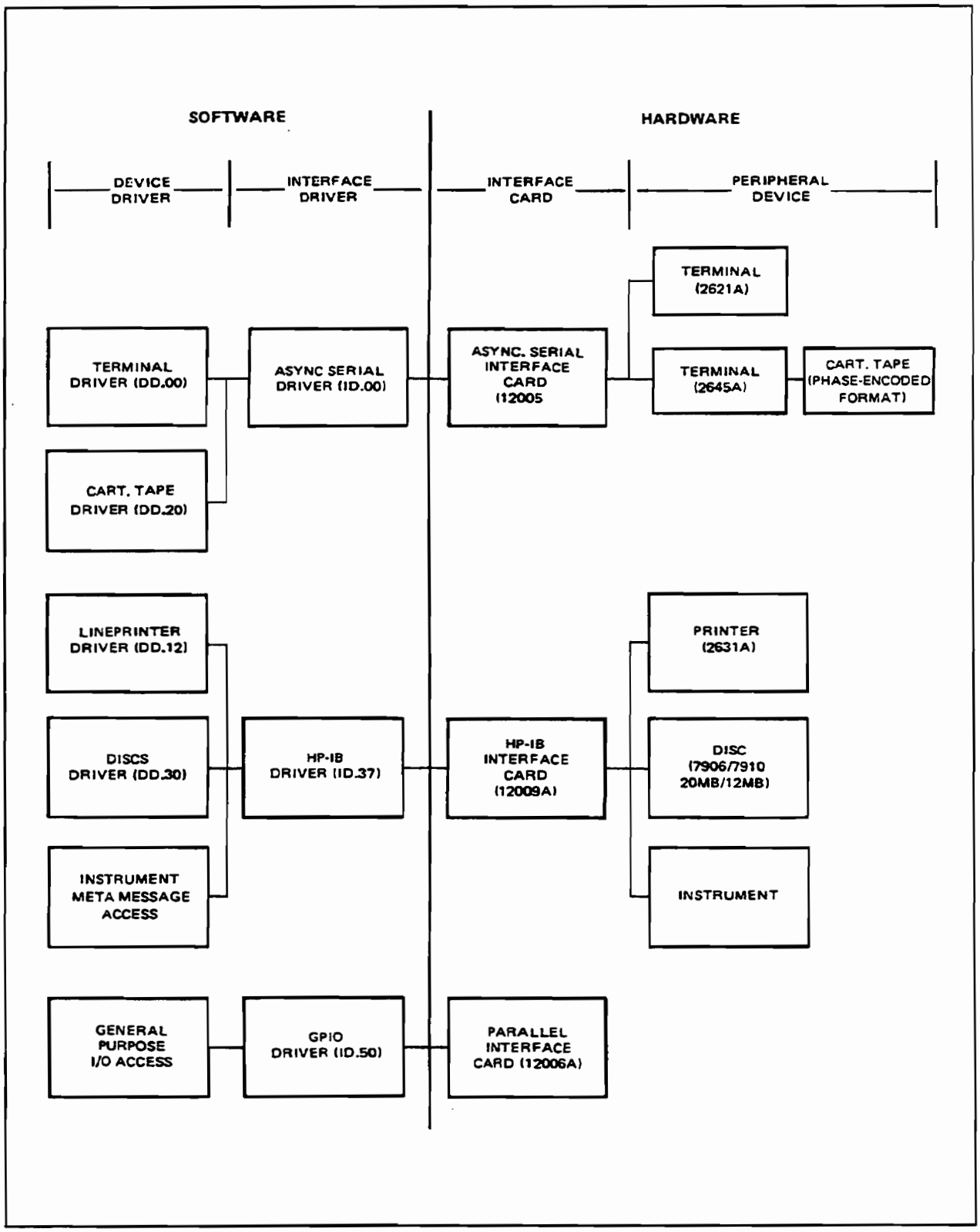


Figure 1-4. Software/Hardware Interface

I/O Master

The I/O Master is the standard interface circuitry to the A-Series backplane and is used on every I/O card. It interfaces the device specific circuit on an interface card to the backplane (see Figure 1-5).

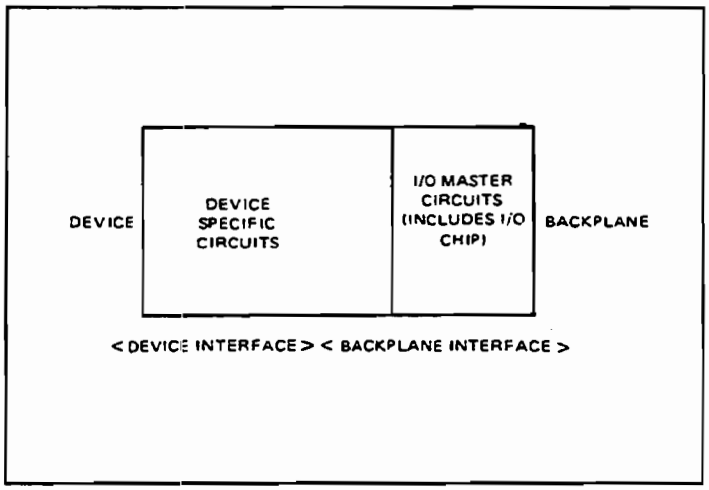


Figure 1-5. I/O Interface Layout

The I/O Master manages most of the I/O functions from the individual I/O cards to free the Central Processor from the detailed management of the I/O functions and to improve system performance. The I/O Master also implements the DMA function on each I/O card. This allows each I/O card to have access to memory and to manage its own transfers to and from memory. The I/O Master contains the I/O chip and associated logic circuits that control all interactions between the processor, main memory, and the I/O card. This includes decoding and executing I/O instructions and DMA functions.

The A-Series I/O system features a multilevel, vectored priority interrupt structure. There are 60 distinct interrupt levels, only 47 of which are user assignable, each interrupt having a unique priority assignment. Any I/O device can be selectively enabled or disabled using the Interrupt Mask Register feature, or the entire interrupt system (except for unimplemented instruction interrupts) can be enabled or disabled under program control.

User Interface Requirements

This manual assumes that the user wishes to interface a device which is not a standard peripheral supplied by Hewlett-Packard. To operate a user's device with the A-Series Computer, the user must consider what software driver, interface card, and cable will be used. This manual will aid the user in choosing from available HP interface products or in designing his own interface products.

There are several possible methods of accomplishing these objectives. For the hardware, you may be able to use an available HP general purpose interface card or design and build a custom

interface card. For software, writing a short assembly-language subroutine may suffice, or full Real-Time Executive drivers may have to be written. As described previously, the A-Series driver structure consists of two parts, the interface driver and the device driver. For further software development information, refer to the applicable software system documentation listed in Table 1-2.

Levels of Hardware Interfacing

For the purposes of this manual, the approaches to interfacing break down into three levels. Note that all levels of interfacing to a device require the proper cabling to be defined.

Level 1 – Using existing HP I/O interface cards and drivers;

Level 2 – Using existing HP I/O interface cards with custom Device Drivers; and

Level 3 – Using custom I/O interface cards designed using the I/O Master with custom Interface and Device Drivers.

Level 1

Assumes that the specifications of existing I/O interface cards and Interface Drivers match the user's device. These interface cards cover a wide range of applications. A discussion of these interface cards is contained in Chapter 3. In addition, data sheets describing the features and specifications of the I/O interfaces, and a list of product support documentation and software are available from the nearest HP Sales and Service Office.

Level 2

Assumes that the specifications of the interface cards and interface drivers are satisfactory to operate your devices, but device drivers must be modified or created. For example, a special HP-IB device might be interfaced using the HP-IB card, the HP-IB interface driver and by writing a custom device driver. The interface and device drivers are described in the *RTE-A Driver Reference Manual,* part number 92077-90011.

Level 3

This is the most advanced level of interfacing. The I/O Master simplifies design, allowing the designer to concentrate on device interfacing rather than backplane interfacing. The I/O Master physical layout and the use of a six layer PC card combine to reduce cross-talk and provide proper shielding for high speed data transfers. Thus, this manual assumes the designer will use the I/O Master to interface to the A-Series. The I/O Breadboard card (HP 12010A) consists of the I/O Master and provides space for circuit prototyping. Once the design is complete, manufactured cards may be built using the layout masks for the I/O Master.

At all interfacing levels, the user should keep in mind that Hewlett-Packard warranties and responsibilities apply only to those items produced and quality controlled by Hewlett-Packard.

This manual is intended only as a guide, and the effectiveness of devices or programs created according to the recommendations outlined herein are purely the responsibility of the user.

Available Documentation

Supporting hardware documentation is provided with each Hewlett-Packard computer. Hardware documentation is also supplied for optional and accessory add-ons as well as for standard I/O interface cards. Basic hardware manuals for the HP 1000 A-Series Computer are listed in Table 1-2; hardware manuals for the I/O cards are listed in Chapter 3 of this manual.

All software supplied with an HP computer system is supported by complete user documentation. Software manuals include language manuals, operating system manuals, and user manuals. Reference manuals that contain basic information for writing system software drivers are listed in Table 1-2. Consult the nearest HP Sales and Service Office for additional hardware or software documentation related to the A-Series computer.

Table 1-2. I/O Interfacing Related Reference Manuals

HP 1000 A-SERIES COMPUTERS	
TITLE	**HP PART Number**
HARDWARE	
HP 1000 A-Series Computer I/O Interfacing Guide	02103-90055
HP 1000 A400 Computer Reference Manual	02424-90001
HP 1000 A600+ Computer Reference Manual	02156-90001
HP 1000 A700 Computer Reference Manual	02137-90001
HP 1000 A900 Computer Reference Manual	02139-90001
HP 1000 A990 Computer Reference Manual	02959-90001
HP 1000 Model 26/27/29 Computer System Installation and Service Manual	02196-90002
HP 1000 A400 Computer Installation and Service Manual	02134-90001
HP 1000 A600+ Computer Installation and Service Manual	02156-90002
HP 1000 A700 Computer Installation and Service Manual	02137-90002
HP 1000 A700 User Control Store Installation and Reference Manual	02137-90003
HP 1000 A900 Computer Installation and Service Manual	02139-90002
HP 1000 A990 16-Slot System/Computer Installation and Service Manual	02939-90001
HP 1000 A990 20-Slot System/Computer Installation and Service Manual	02959-90002
HP 1000 A990 Upgrade Installation and Service Manual	12990-90011
HP 1000 A990 Computer Reference Manual	02959-90001
HP 1000 Micro 24/26/27/29 System/Computer Installation and Service Manual	02430-90010
HP 12156A Floating Point Processor Kit Installation and Reference Manual	12156-90001
HP 1000 A400 Engineering and Reference Documentation	02424-90003
HP 1000 A600 Engineering and Reference Documentation	02156-90003
HP 1000 A700 Engineering and Reference Documentation	02137-90005
HP 1000 A900 Engineering and Reference Documentation	02139-90003

Table 1-2. I/O Interfacing Related Reference Manuals (Continued)

HP 1000 A-SERIES COMPUTERS	
TITLE	HP PART Number
SOFTWARE	
HP 92049A Microprogramming Package Reference Manual	92049-90001
RTE-A Debug Reference Manual	92077-90010
RTE-A Driver Reference Manual	92077-90011
RTE-A System Design Manual	92077-90013
RTE-A Driver Designer's Manual	92077-90019
RTE-A Generation Requirements for Drivers	92077-90021
RTE-A EXEC Control Technical Specifications Manual	92077-90022
RTE-A I/O Control Technical Specifications Manual	92077-90023
RTE-A Drivers Technical Specifications Manual	92077-90027

2

Programming

I/O System Programming

This section provides specific code segments that perform each of the functions of the I/O system and any special considerations involved. This should greatly simplify the task of the designer in determining when a new driver is needed or when an existing driver needs to be modified. It is not mandatory that these code segments be used as presented, but the purpose of each segment must be fully carried out. For additional information, refer to Table 1-2 for related reference manuals.

System Level I/O (Select Code 0-17)

Included as standard features on the A-Series central processor card is the circuitry for the following functions:

a. Input and output of several system information status bits;

b. Enabling and disabling of the interrupt system;

c. Enabling and disabling of the user ROM mode;

d. Loading, enabling, and disabling of the Global Register;

e. Initiating the Virtual Control Panel feature;

f. Detection and interrupt generation for powerfail/autorestart;

g. Identification of the interrupting device via the Central Interrupt Register (CIR);

h. Enabling, disabling, and interrogating (for the appropriate address) the memory protect and the parity error logic;

i. Starting and stopping the Time Base Generator (TBG);

j. Enabling, detecting, and interrupting on an unimplemented instruction opcode.

Each of these functions is associated with a select code in the range 0–17 octal and each select code is dealt with separately in the following paragraphs.

Interrupt System, Control Reset, and Interrupt Mask (Select Code 0)

The state of the interrupt system (enabled or disabled) is controlled by the Flag flip-flop associated with select code 0. An STF 0 enables level three interrupts (I/O and TBG) and a CLF 0 disables level three interrupts. The status of the interrupt system may be interrogated by the instructions SFS 0 and SFC 0, which skip if the level three interrupts are enabled or disabled, respectively.

The CLC 0 instruction generates the backplane signal CRS (Control Reset), which completely resets the I/O system. This serves to halt all I/O activity, including DMA.

The STC 0 instruction is a NOP.

An additional feature is the interrupt mask, which allows selective disabling of the interrupt capability in groups of four select codes (SC). This is accomplished by executing an OTA/B 0, where the bits of A/B are as follows:

BIT	SC INTERRUPT DISABLES (when set)
15	77–74
14	73–70
13	67–64
12	63–60
11	57–54
10	53–50
9	47–44
8	43–40
7	37–34
6	33–30
5	27–24
4	23–20
3	17–14 (reserved for CPU use)
2	13–10 (reserved for CPU use)
1	6 (TBG)
0	STATUS

Any subsequent interrupt requests that the interface would like to generate are held off until its associated bit is reset.

The current status of the interrupt mask may be read by setting the Global Register to any existing interface select code and executing an LIA/B 0. The selected interface will put its copy of the interrupt mask into A/B. Each bit will have the meaning shown above, with additional information contained in bit 0. Bit 0 will indicate whether this card's interrupt mask bit is in the set (1) or reset (0) condition. An MIA/B 0 performs the same function except that the mask status is merged into A or B.

Status Register, Processor LEDs, Switches and Overflow Bit (Select Code 1)

The overflow bit is the flag associated with select code 1. An STF 1 or STO instruction sets the overflow bit (as does an overflow from an appropriate arithmetic instruction); a CLF 1 (or CLO) instruction clears it. The SFS 1 (or SOS) and SFC 1 (or SOC) instructions skip if overflow is set or clear, respectively.

Instructions referencing select code 1 do not cause memory protect violations (except HLT 1).

The processor LEDs respond to an OTA/B 1. The lower byte of the A/B-Register is written into the register by the OTA/B 1 instruction. A lit LED corresponds to a logical 1 level.

The processor startup switches are read by an LIA/B 1 (upper 8 bits only). The switch, bit, and function relationship are as follows:

SWITCH	BIT	MEANING
S1	8	Boot select
S2	9	Boot select
S3	10	Boot select
S4	11	Boot select
S5	12	VCP Program select
S6	13	Not used
S7	—	A990 only; must be down
S8	14	MEM lost
—	15	A600 only: TBG (Interrupt mask bit 1)

Global and Diagnose Registers, and ROM Enable (Select Code 2)

INSTRUCTION	FUNCTION
OTA/B 2	Configure Global Register to the value contained in the A/B-Register.
CLF 2	Enable Global Register.
STF 2	Disable Global Register.

The above instructions are for use during normal system operation. The I/O chip has implemented some additional functions involving select code 2 which may be useful for system configuration.

Each I/O chip has two registers, the Global Register itself and a Diagnose Register, which are accessed by referencing select code 2. The Global Register is stored to by execution of an OTA/B 2 instruction with the value of A/B greater than or equal to 10 (octal).

The Diagnose Register is stored to by execution of an OTA/B 2 instruction with the value of the A/B-Register less than 10 (octal). The value stored into the Diagnose Register selects one of eight possible special operating modes.

Diagnose mode 0 (0 stored in the Diagnose Register) is the normal state of the IOP chip. The chip powers up in this mode, and should be returned to this mode after any diagnostics.

If the value stored into the Diagnose Register was 1, the interrupt priority chain is used to cause each I/O interface to respond to an LIA/B 2 in sequence, where the sequence is determined by the physical proximity to the CPU on the interrupt priority chain, that is, the closest interface responds to the first LIA/B 2, the next closest to the next LIA/B 2, and so on. After the lowest priority interface has responded to its LIA/B 2, subsequent LIA/B 2s will become no-ops, leaving A/B unchanged. The data that is transferred by each I/O card has the following format:

BIT	MEANING
15	If set, the card is an intelligent interface. It can perform its own self-test.
14–9	Interface identification number—used to identify one type interface card from other types of interface cards.
8–6	Interface revision level.
5–0	Interface select code.

This mode can be used to determine which types of interfaces are present, what their select codes are, and their positions relative to each other.

If the value stored into the Diagnose Register was 2, the interrupt priority chain will again be used to sequence through the interfaces present in the system. The information that is transferred during each LIA/B 2 is as follows:

BIT	MEANING
15–13	HP Reserved.
12	If set, break feature is enabled for this interface.
11	Priority in. If set, this interface has the interrupt priority signal asserted from the interface above it.
10	Flag 23 state.
9	Flag flip-flop state.
8	Control flip-flop state.
7	If set, indicates that the Global Register contents equal this interfaces select code.
6	If set, the Global Register is enabled.
5–0	Current Global Register value as stored by this interface.

Either of the above modes may be terminated by executing an OTA/B 2 with the value of A/B equal to 0; the mode may be restored by again storing the value 1 or 2 (as appropriate) into the Diagnose Register (this will start the sequence over again with the highest priority card).

If the value stored into the Diagnose Register is 3, a DMA diagnostic sequence is initiated in which the DMA logic writes the contents of Register 20 into the address register (22) and increments both registers. This operation repeats until the count in Register 23 is exhausted.

Note

The DMA logic must be properly configured for a word mode input transfer before Diagnose mode 3 is entered.

Storing the values 4–6 into the Diagnose Register does not invoke any special mode.

Storing the value 7 into the Diagnose Register asserts the SRQM+ (Service Request Mask) signal to the interface function. This causes the I/O chip to ignore all assertions of SRQ (Service Request) by the interface.

An STC 2 is used to enable the break feature of the I/O device configured for VCP operation. CLC is a NOP.

Virtual Control Panel (Select Code 3)

The Virtual Control Panel (VCP) can be invoked by executing a HLT instruction, if an I/O Interface card has been selected to function as the VCP. The HLT executes as a NOP if no card is configured as the VCP.

An OTA/B 3 stores the contents of the A/B-Register in the P-save register in the I/O chip when enabled for VCP. Conversely, an LIA/B 3 loads the contents of the P-save register into the A/B-Register when the interface is enabled for VCP and a BRK signal is issued. An OTA/B 3,C is used to store the starting address of the VCP program into the P-save register; then an LIA/B 3,C will load the A/B-Register with the starting address of the VCP program when the interface is enabled for VCP and a BRK is issued.

Powerfail/Autorestart and CIR (Select Code 4)

The powerfail interrupt is not the highest priority interrupt in the system and can be temporarily held off by higher priority interrupts that must be serviced immediately to be interpreted correctly or to preserve the integrity of memory. If, for example, both parity error and powerfail try to interrupt simultaneously, parity error will get serviced first. However, powerfail has the highest processing priority (that is, in the above situation the parity error interrupt would occur first, but as soon as the instruction in the trap cell finished executing, the powerfail interrupt would occur). powerfail interrupts (and all lower interrupts) will be held off until after the execution of the next instruction if the processor card asserts TDI (Temporarily Disable Interrupt). TDI– is asserted for 10 group instructions, JMP,I, JSB,I, and for enabling the bootstrap ROMs in slave mode processing. This is done to eliminate complications that would arise in saving the state of the system if an interrupt occurred during one of these instructions. This short delay in powerfail service is more than compensated by the resulting simplification of powerfail interrupt routines. It is advisable to separate the entry point for the powerfail interrupt from that used for any other interrupt.

The Central Interrupt Register (CIR) is loaded with the select code of the device which caused the last interrupt. It is loaded during the fetch of the trap-cell instruction, when the interface select code is being used to address memory.

The select code 4 instructions are as follows:

INSTRUCTION	MEANING
STC 4/CLC 4	Either of these instructions serves to re-initialize the powerfail logic and to enable and disable level 2 and 3 interrupts, respectively.
STF 4/CLF 4	None. The flag of this select code indicates the CLF 4 status of the power supply; set means power is stable and clear means power is failing.
SFS 4	Skip if power is not going down.
SFC 4	Skip if power is failing.
LIA/B 4	Loads the contents of the CIR into A/B.
OTA/B 4	Write to the CIR the contents of A/B.

Note that an autorestart interrupt at location 00004 occurs only if that location's contents are not zero; otherwise, the system is rebooted. This is done so that if powerfails and is restored during a boot, an attempt to restart a partially loaded program can be avoided. To enable this to happen, the program being loaded should initially load location 00004 with zero and load the powerfail JSB instruction only when the load is otherwise complete.

Parity Error (Select Code 5)

The parity error feature of the memory detects when a single-bit or odd number of bits in a word fetched from memory is incorrect, and, if enabled, will generate a parity error interrupt request.

A parity error stores the address of the word that had the incorrect bit.

The following instructions apply to select code 5:

INSTRUCTION	MEANING
STC 5	Enable parity interrupts.
CLC 5	Disable parity interrupts.
STF 5	Select even parity.
CLF 5	Select odd parity.
SFS 5	Skip if even parity.
SFC 5	Skip if odd parity.
LIA/B 5	Load parity error address from parity register into A/B. (This is bits 0 through 15 of physical address.)

LIA/B 5,C	Load parity error address (bits 16 through 23 of physical address) into lower byte of A/B.

Time Base Generator (Select Code 6)

The built-in Time Base Generator (TBG) is a timer that requests an interrupt every 10 milliseconds, once it is started. The first interrupt request occurs 10 milliseconds after the TBG is turned on, with subsequent interrupt requests occurring every 10 milliseconds.

The pertinent instructions are as follows:

INSTRUCTION	EFFECT
STC 6	Enable the time base generator tick.
CLC 6	Disable the time base generator tick.
STF 6	Set TBG flag.
CLF 6	Clear TBG flag.
SFS 6	Skip if TBG flag is set.
SFC 6	Skip if TBG flag is clear.
LI*/MI*	with select code 6 are NOPs.
OT* 6	Set TBG frequency (A990 only).

* = A or B

Memory Protect and Violation Register (Select Code 7)

The memory protect feature of the processor, when configured and enabled, does not allow execution of most I/O instructions (select code 1 may be referenced), and does not allow the currently executing program to:

a. Jump to an address in a protected page.

a. Jump to a subroutine in a protected page, unless the address is 0 or 1 (A- or B-Register).

a. Store into an address in a protected page, unless the address is 0 or 1, the A- or B-Register.

Memory is mapped and protected on a page-by-page basis.

The Memory Protect logic stores the address of the instruction that caused the memory protect violation in the violation register.

The following instructions apply to select code 7:

INSTRUCTION	MEANING
STC 7	Enable the memory protect feature.
CLC 7	NOP.

STF 7	NOP.
CLF 7	NOP.
SFS 7	NOP.
SFC 7	NOP.
LIA/B 7,C	Load the contents of the Violation Register into A/B.
MIA/B 7,C	NOP.
OTA/B 7	NOP.
OTA/B 7,C	NOP.

Unimplemented Instruction (Select Code 10)

A feature unique to the A-Series is the capability to generate an interrupt on detection of the attempted execution of a currently unimplemented instruction. The unimplemented instruction trap is always active (it can never be turned off). When an unimplemented instruction is recognized, it generates an interrupt through select code 10.

Reserved (Select Code 11-16)

These select codes are reserved for possible future use by Hewlett-Packard.

Interrupt Priority

There are two types of interrupt requests in the A-Series Computer, system level interrupts and I/O interrupts. System level interrupts are generated by the processor card to handle system level problems such as powerfail and parity error. I/O interrupts may be requested by the individual I/O cards to cause processing to service the needs of that I/O interface.

Both system level and I/O interrupts are handled by the processor card. Three basic levels of importance define the priority of interrupt requests. Level one interrupts, parity error and unimplemented instruction, have no restrictions in obtaining interrupt service. Memory protect violation and powerfail are the level two interrupts. They can be collectively enabled/disabled by an interrupt inhibit flag (STC/CLC 4). Level three interrupts, time base generator and all I/O interrupts, can be enabled/disabled by a STC/CLC 4 or an interrupt system flag (STF/CLF 0). The following chart shows pictorially the relative priority and the qualifiers required by each interrupt request.

level 1
parity error during a CPU memory access
unimplemented instruction
**********interrupt inhibit flag (STC/CLC 4)

level 2
memory protect violation
Temporarily Disable Interrupt (TDI–)*
powerfail
**********interrupt system flag (STF/CLF 0)

level 3
----------interrupt mask (OTA/B 0)
time base generator
I/O interrupt requests

* The Temporarily Disable Interrupt (TDI–) signal is utilized to resolve complications that would arise if an interrupt occurred while executing an indirect jump instruction. For the next instruction cycle following a jump, indirect (JMP,I); a jump to subroutine, indirect (JSB,I); an I/O group instruction; or enabling the bootstrap ROMs in slave mode processing; the processor will hold off the powerfail interrupt request and all level three requests. Up to three levels of indirect jumps will keep these requests disabled.

Instruction Usage Summary

Table 2-1 lists all the instructions recognized by the IOP chip, by their select code. Consult the appropriate interface card reference manual for details on how each of these instructions is interpreted by the specific interface.

Table 2-2 lists the processor card instructions for select codes 00 through 07. Some instructions appear in both Table 2-1 and Table 2-2 with a different description in the function column of each table, for example, STC 2 is described as "Enable Slave logic" in Table 2-1, and "Disable Boot ROM" in Table 2-2. This means that the instruction is executed by the IOP chip to enable the slave logic, and simultaneously the processor is executing the instruction to disable the ROM loader.

Interface I/O (Select Codes 20-77)

A-Series I/O interfaces are capable of transferring data directly to or from memory (direct memory access) or through the central processor A- and B-Registers (programmed I/O). A detailed explanation of each method will be given in the following paragraphs.

Direct Memory Access

The Direct Memory Access (DMA) feature of every A-Series I/O Interface Card allows input or output operations to proceed without processor intervention, significantly easing the processing requirement on the CPU. DMA transfers are also useful for high-speed devices such as disks that could not tolerate a long interrupt latency when transferring data. The use of DMA to perform I/O data transfers reduces the number of interrupts to the CPU from one per byte or word to one per complete DMA block transfer. The maximum DMA block size is 65,536 bytes.

The maximum DMA transfer rate and bandwidth of the backplane given in the specifications of Table 1-1 is the cumulative rate that multiple devices can perform concurrent DMA transfers. When DMA transfers are operating at full bandwidth, the central processor can interleave instruction fetch memory cycles with DMA transfers at the rate of one in every 32 memory accesses for A700 computers. For A600+ computers, DMA may completely monopolize the backplane and consequently deny the processor access to memory. For A900 and A990 computers, DMA can take over the entire I/O backplane without stopping the CPU. This is because the A900 processor does not make memory reference operations across the I/O backplane.

Any given interface has a maximum rate at which it is capable of running. In order to achieve this maximum rate, however, there can be no other higher priority requests on the backplane which degrade the maximum possible DMA rate. When the central processor decides to fetch an instruction just before an interface card requests a DMA transfer, when RNI is asserted that DMA cycle can be held off for 908 nanoseconds for the A600+, 1.0 microsecond for the A700, and 1088 nanoseconds for the A900 and A990. This occurs only on I/O instruction fetches.

If the interface is in the highest priority slot, it can be held off by the central processor, or, in addition, by a lower priority interface doing a DMA transfer, for a worst case of 227 nanoseconds for the A600+, 250 nanoseconds for the A700, and 534 nanoseconds for the A900 and A990. This worst case would occur when the higher priority interface doesn't have another word ready for transfer at the end of the last memory access, in which case the next highest priority requester would be granted access to memory. This memory access will complete regardless of other higher priority requests, holding them off until after it is through. All DMA specifications in the preceding paragraph and in Table 1-1 assume the interface in question is plugged into the highest priority slot. When plugged into a lower priority slot, DMA rates cannot be guaranteed when the cumulative rates of the higher priority cards are capable of consuming the total backplane bandwidth. Typically, however, even in a busy system, the backplane is only congested for very short periods of time, so that even the lowest priority interface cards transfer at rates very close to their normal specifications.

The central processor has lower priority than any I/O card, and could be held off indefinitely, however, the signal CPUTURN is used to prevent this occurrence in the A700 and A900 computers. Whenever the processor needs access to the backplane and an interface is executing, a memory cycle counter in the processor is started. If the processor is unable to gain access to the backplane within 32 memory cycles the signal CPUTURN will be asserted, and the processor will have access to the backplane for one memory cycle before further DMA cycles. This guarantees the processor approximately one out of every 32 memory cycles in a congested backplane, allowing interrupt latency to be guaranteed. For the A900, if a processor request for the backplane is blocked by MRQ (DMA), the processor will immediately assert CPUTURN.

Table 2-1. IOP Chip Instructions By Select Code

Instruction	Function	Notes
LI* 0	Read interrupt mask	
MI* 0	Merge interrupt mask	
OT* 0	Write interrupt mask	
CLF 2	Enable Global Register (GR)	
STF 2	Disable GR	
LI* 2 [, C]	Read GR	1, 10
MI* 2 [, C]	Merge GR	1, 10
OT* 2 [, C]	Write GR	1
STC 2 [, C]	Enable Slave logic	1, 2
[, C]	Enable GR	
HLT 3, C	VCP call back (used in boot only)	2
LI* 3	Read P SAVE	2
MI* 3	Merge P SAVE	2
OT* 3	Write P SAVE	2
LI* 3,C	Read ROM P	2
MI* 3,C	Merge ROM P	2
OT* 3,C	Write ROM P	2
SFC 20	Skip if FLG 20 clear (skip if self-configured DMA transfer still in progress)	3
SFS 20	Skip if FLG 20 set (skip if self-configured DMA transfer completed)	3
CLF 20	Clear FLG 20 and FLG 21	3
STF 20	Set FLG 20	3
STC 20 [, C]	Enable DMA self configuration	3
CLC 20 [, C]	Disable DMA self configuration	3
LI* 20 [, C]	Read DMA configuration address	3
MI* 20 [, C]	Merge DMA configuration address	3
OT* 20 [, C]	Write DMA configuration address	3
[, C]	Clear FLG 20 and FLG 21	3
SFC 21	Skip if FLG 21 clear (skip if standard DMA transfer still in progress)	4
SFC 21	Skip if FLG 21 set (skip if standard DMA transfer completed)	4
CLF 21	Clear FLG 21	4
STF 21	Set FLG 21	4
STC 21 [, C]	Enable DMA transfers	4
CLC 21 [, C]	Disable DMA transfers	4
LI* 21 [, C]	Read DMA Control word	4
MI* [, C]	Merge DMA Control word	4
OT* 21 [, C]	Write DMA Control word	4
[, C]	Clear FLG 21	4

Table 2-1. IOP Chip Instructions By Select Code (Continued)

Instruction	Function	Notes
SFC 22	Skip if FLG 22 clear (skip if no parity error during DMA read)	5
SFS 22	Skip if FLG 22 set (skip if parity error during DMA read)	5
CLF 22	Clear FLG 22	5
STF 22	Set FLG 22	5
CLC 22 [, C]	Force DMA reconfiguration	5
LI* 22 [, C]	Read DMA address	5
MI* 22 [, C]	Merge DMA address	5
OT* 22 [, C]	Write DMA address	5
[, C]	Clear FLG 22	
SFC 23	Skip if FLG 20, FLG 21, AND FLG 22 ALL Clear	6
SFS 23	Skip if FLG 20 OR FLG 21 OR FLG 22 Set (inclusive OR)	6
CLF 23	Clear FLG 20, FLG 21, and FLG 22	6
CLC 23 [, C]	Terminate DMA operation	6
LI* 23 [, C]	Read DMA Count	6
MI* 23 [, C]	Merge DMA Count	6
OT* 23 [, C]	Write DMA Count	6
[, C]	Clear FLG 20, FLG 21, and FLG 22	6
SFC 24	Skip if DMA disabled (DMAEN--not asserted)	
SFS 24	Skip if DMA enabled (DMAEN--asserted)	
LI* 24	Read Scratch 1	
MI* 24	Merge Scratch 1	
OT* 24	Write Scratch 1	
LI* 25	Read Scratch 2	
MI* 25	Merge Scratch 2	
OT* 25	Write Scratch 2	
LI* 26	Read Scratch 3	
MI* 26	Merge Scratch 3	
OT* 26	Write Scratch 3	
SFC 30	Skip if FLG 30 clear (skip if data transfer still in progress)	7
SFS 30	Skip if FLG 30 set (skip if data transfer completed)	7
CLF 30	Clear FLG 30	7
STF 30	Set FLG 30	7
STC 30 [, C]	Set CNTRL 30 and issue DVCMD	7
CLC 30 [, C]	Clear CNTRL 30	7
LI* 30 [, C]	Read device data	7
MI* 30 [, C]	Merge device data	7
OT* 30 [, C]	Write device data	7
[, C]	Clear FLG 30	

Table 2-1. IOP Chip Instructions By Select Code (Continued)

Instruction	Function	Notes
LI* 31	Read interface control word	
MI* 31	Merge interface control word	
OT* 31	Write interface control word	
LI* 32-77	Read user aux	8
MI* 32-77	Merge user aux	8
OT* 32-77	Write user aux	8, 9

* = A- or B-Register

NOTES:

1. The [,C] is always executed even if the primary instruction is not.
2. The SLAVE logic must be enabled: that is, SLRQ LOW at power up.
3. FLG 20 is set by DMA upon completion of self-configured DMA block transfer which is not to be followed by another self configuration.
4. FLG 21 is set by DMA upon completion of any block transfer which is not to be followed by a self configuration.
5. FLG 22 is set by DMA if a parity error occurs during a DMA read.
6. FLG 23 is the logical OR of flags 20 through 22.
7. FLG 30 and CNTRL 30 are controlled by the Flag and Control flip-flops located in the IOP chip.
8. The IOP chip indicates only that the select code is in the range 32 to 77, it is up to the user to decode any specific select code.
9. Several I/O cards, by convention, use this instruction as a card reset.
10. When the IOP chip is in diagnose mode, these instructions fetch the following:

Diagnose mode 1---the interface's select code and ID word.
Diagnose mode 2---the Global Register and IOP chip status bits.

DMA Control Words

All control logic and registers necessary to supervise the memory transaction are contained in the IOP chip. Appropriate signals are available to the interface logic to signal when to enable its data onto the system data bus, when to receive data from the system data bus, and when to initiate the next interface operation. In addition, signals to the IOP chip indicate when a data transfer is needed and when an interrupt-requesting condition has occurred.

Software DMA setup and control involves four different control words, each of which is associated with a different register. The four registers are control and status registers located in the IOP chip.

Three of these registers must be loaded with control words that specify the DMA operation. The fourth register is used for a special type of DMA operation called self-configured DMA, which is diskussed later. The DMA registers and their functions are as follows:

a. Register 20, DMA Self-Configuration Register;

b. Register 21 (for Control Word 1), DMA Control Register;

c. Register 22 (for Control Word 2), DMA Address Register; and

d. Register 23 (for Control Word 3), Word/Byte Count Register.

Control Word 1—Initialization

Control Word 1 (CW1) must be loaded into Register 21 of the desired I/O interface as part of the DMA initialization process. The general definitions of the bits in Control Word 1 are given in Figure 2-1. Note that the requirements of individual I/O interfaces may vary slightly from the general definitions. Refer to the I/O interface reference manuals for specific programming information.

Control Word 2—Buffer Address

Control Word 2 (CW2) loads into Register 22 the address of the first memory location to be read from or stored into when the DMA operation is initiated. The most significant bit, bit 15, is not used by the DMA control logic.

Control Word 3—Word/Byte Count

Control Word 3 (CW3) loads into Register 23 the twos complement number of data elements to be transferred by DMA. Data elements may be either words or bytes as specified by bit 13 of CW1 (Figure 2-1). On each transfer the count register is incremented. The end of a DMA transfer is indicated by the transition from -1 to 0 of the value in Register 23. This transition causes the I/O interface to generate a completion interrupt, if completion interrupts are enabled by bit 11 of CW1 (Figure 2-1).

Standard DMA Transfer Initialization

A DMA data transfer is started by:

a. Loading the Global Register with the select code of the desired I/O interface;

b. Issuing a CLC instruction to Register 21 (DMA Control Register) to initialize DMA operation;

c. Loading the three DMA registers;

d. Loading the control register (Register 31) of the I/O interface (described in the individual I/O interface reference manuals); and

e. Issuing an STC instruction to Register 21 (DMA Control Register) to start DMA operation.

Table 2-2. Processor Card Instructions for Select Codes 00 through 07

Instruction	Function
STC 0	NOP
CLC 0	System reset
STF 0	Enable Type 3 interrupts
CLF 0	Disable Type 3 interrupts
SFS 0	Skip if Type 3 interrupts enabled
SFC 0	Skip if Type 3 interrupts disabled
LI* 0	Load from interrupt mask register
MI* 0	NOP
OT* 0	Output to interrupt mask register
STC 1	NOP
CLC 1	NOP
STF 1	Same as Set Overflow (STO)
CLF 1	Same as Clear Overflow (CLO)
SFS 1	Same as Skip If Overflow Set (SOS)
SFC 1	Same as Skip If Overflow Clear (SOC)
LI* 1	Load from processor card status register
MI* 1	Merge from processor card status register
OT* 1	Output to processor card status register
STC 2	Enable break feature
CLC 2	NOP
STF 2	Disable global register
CLF 2	Enable global register
SFS 2	Skip if global register disabled
SFC 2	Skip if global register enabled
LI* 2	Load from global register
MI* 2	NOP
OT* 2	Output to global register (Note 1)
STC 3	NOP
CLC 3	NOP
STF 3	NOP
CLF 3	NOP
SFS 3	NOP
SFC 3	NOP
LI* 3	Load from P SAVE
MI* 3	NOP
OT* 3	Output to P SAVE
LI* 3, C	Load from ROM P
MI* 3, C	NOP
OT* 3, C	Output to ROM P
STC 4	Enable Type 2 and 3 interrupts
CLC 4	Disable Type 2 and 3 interrupts

* = A or B.
Note 1. An OTA/B 2 with A/B equal to one through seven establishes a diagnose mode.

Table 2-2. Processor Card Instructions for Select Codes 00 through 07 (Continued)

Instruction	Function
STF 4	NOP
CLF 4	NOP
SFS 4	Skip if power is stable
SFC 4	Skip if power going down
LI* 4	Load from central interrupt register
MI. 4	NOP
OT* 4	Output to central interrupt register
STC 5	Enable parity error interrupts
CLC 5	Disable parity error interrupts
STF 5	Set parity sense to even parity
CLF 5	Clear parity sense to odd parity
SFS 5	Skip if parity sense is even
SFC 5	Skip If parity sense is odd
LI* 5	Load bits 0 through 15 from parity register
LI* 5, C	Load bits 16 through 23 from parity register
MI * 5	NOP
OT* 5	Set ECC bits (A990 only)
STC 6	Turn on time base generator
CLC 6	Turn off time base generator
STF 6	Set time base generator flag
CLF 6	Clear time base generator flag
SFS 6	Skip if time base generator flag set
SFC 6	Skip if time base generator flag clear
LI* 6	NOP
MI* 6	NOP
OT* 6	Set TBG frequency (A990 only)
STC 7	Enable memory protect
CLC 7	NOP
STF 7	NOP
CLF 7	NOP
SFS 7	NOP
SFC 7	NOP
LI* 7	Load from violation register
MI * 7	NOP
OT* 7	Output to memory protect fence register
LI* 7, C	NOP
MI* 7, C	NOP
OT* 7, C	NOP

* = A or B.
Note 1. An OTA/B 2 with A/B equal to one through seven establishes a diagnose mode.

A typical programming sequence to configure the DMA logic for a DMA transfer is as follows:

```
LDA SC          Load select code
OTA 2,C         Set up Global Register
CLC 21B         Initialize DMA
LDA CW1
OTA 21B         Output DMA control word
LDA CW2
OTA 22B         Output DMA starting address
LDA CW3
OTA 23B         Output DMA word/byte count
LDA CNT
OTA 31B         Output I/O interface control word
STC 21B,C       Start DMA
<continue any other processing>
```

Note that the Global Register is needed only for accessing the various registers on the card, and can be reconfigured after starting up DMA without affecting the DMA transfer.

Standard DMA Data Transfer

Figure 2-2 illustrates, in general, the sequence of operations for a DMA input data transfer (the minor differences for an output transfer are explained in text). Note that the Global Register has been enabled and loaded with the I/O card's select code.

The initialization routine sets up the DMA control registers on the I/O card (1) and issues the start command (STC 21,C) to the DMA Control Bit (Control 21). (If the operation is an output, the I/O card buffer is also loaded at this time.) The DMA logic is now turned on and the computer program continues with other instructions.

Setting the DMA Control bit (2) causes the I/O card to send a Start signal (with a data word if it is an output transfer) to the external device (3). The device goes through a read or write cycle and returns a Done signal (with a data word if it is an input transfer). The Done signal (4) requests the DMA logic (5) to transfer a word into (or out of) memory (6). The process now loops back to step 3 to transfer the next word.

After the specified number of words have been transferred, the DMA logic generates a completion interrupt (7). The program control is now passed to a completion routine (8), the content of which is the programmer's responsibility. Note that one responsibility of the completion routine is to reset the DMA logic of the I/O interface by clearing Control 21.

Standard DMA Transfer Termination

A DMA data transfer continues as described above until a terminating condition is detected. The terminating conditions are:

a. Word count transition from -1 to 0. (Rollover from octal 177777 to 000000 occurs.) This indicates that all the data elements that were requested to be transferred have been transferred.

b. End-of-transmission indication by the user interface logic upon recognition of a special character. This terminates the DMA transfer even though the word count to be transferred may not have been reached.

c. Parity error indication from memory during the read requested by the DMA control logic.

d. Detection of a Control Reset (CRS) signal. This signal is generated by the central processor during its power-up sequencing, or by execution of a CLC 0 instruction.

It should be noted that the interface can effectively suspend the DMA transfer operation by not requesting any further transfers.

Self-Configured DMA

Each I/O interface also has logic that can automatically load the DMA registers with the DMA control words from sequential locations in memory. This process is performed by using the I/O interface's Register 20, Self-Configuration Register. Once the DMA self-configuration feature is initialized by setting the value of Register 20 equal to the memory address of the first word of a list of DMA triplets or quadruplets, an STC to Register 20 starts the feature. This STC to Register 20 has the effect of simultaneously setting the control on Registers 20 and 21 (to achieve an initial state for full execution).

A triplet is of the form, DMA control word, DMA transfer address, and word byte count. The triplet words are the words to be loaded into Registers 21, 22, and 23, respectively. A quadruplet is of the form, DMA control word, I/O-interface control word, DMA transfer address, and word/byte count. Bit 9 of the DMA control word (Control Word 1) determines whether a triplet or quadruplet is loaded. A quadruplet need only be used when the I/ O-interface control word must be changed. As each register is loaded, the contents of Register 20 are incremented, leaving it pointing to the memory location whose contents are to be loaded into the next register. See Figure 2-3 for the formats of triplets and quadruplets.

DMA self-configurations can be chained to enable consecutive DMA transfers via the same I/O interface with a minimum of interrupts. If bit 15 of Control Word 1 is a logical 1, the DMA registers will be loaded with the next triplet or quadruplet in memory (as pointed to by Register 20) upon completion of the current DMA block transfer.

When bit 15 (and CW1 bit 11) is a logical 0, the current DMA block transfer is followed by a completion interrupt.

A typical programming sequence to configure the DMA logic for a self-configured DMA transfer is as follows:

```
LDA SC          Load select code
OTA 2, C        Set up Global Register
LDA PNTR
OTA 20B         Point to first item in list
STC 20B,C       Start self-configured DMA
```

The above program assumes that the set of triplets (or quadruplets) has already been loaded into consecutive locations in memory starting at location PNTR,I (that is, location PNTR contains the starting address of the first set of triplets or quadruplets). The self-configuration feature will immediately fetch the first three (or four) control words and initiates the DMA transfer.

15	14	13	12	11	10	9	8	7	6 5	4 0
CONT	DVCMD	BYTE	RES	CINT	REM	FOUR	AUTO	IN	Various	EXT ADDR BUS

CONT (Continue), bit 15.

Bit 15 = 1: Enable a DMA re-configuration upon completion of a self-configured DMA transfer.

Bit 15 = 0: Stop DMA after current transfer.

DVCMD (Device Command), bit 14.

Bit 14 = 1: Issue a Device Command signal for each data element transferred.

Bit 14 = 0: No Device Command signal issued.

BYTE (Byte/word transfer), bit 13.

Bit 13 = 1: Conduct DMA transfer in byte mode.

Bit 13 = 0: Conduct DMA transfer in word mode.

RES (Residue), bit 12.

Bit 12 = 1: Write word/byte count back into memory.

Bit 12 = 0: Word/byte count is not written.

CINT (Completion Interrupt), bit 11.

Bit 11 = 1: Inhibit DMA completion interrupt.

Bit 11 = 0: Request completion interrupt when word/byte count goes from -1 to 0 and bit 15 equals 0.

REM (Remote), bit 10.

Bit 10 = 1: Enable remote (non-standard) memory for DMA transfer.

Bit 10 = 0: Remote memory not enabled.

FOUR (Fetch four control words), bit 9.

Bit 9 = 1: Causes DMA self-configuration to fetch four control words; that is,, three DMA control words and one I/O card control word.

Bit 9 = 0: Fetch three control words for DMA self-configuration.

AUTO (Automatic), bit 8*.

Bit 8 = 1: Initiate first data transfer once DMA is configured to output, without waiting for an SRQ. For input transfers, enable a Device Command signal after the last data element is transferred.

Bit 8 = 0: For output transfers, wait for a Service Request (SRQ) signal before performing the first transfer, and wait for an SRQ after transferring the last word to complete the operation. For input transfers, the last data element is not followed by a Device Command.

IN (Transfer In), bit 7.

Bit 7 = 1: Perform DMA transfer from I/O device to memory.

Bit 7 = 0: Perform DMA transfer from memory to I/O device.

Various, bits 5 and 6, User definable. These bits are not available in the self-configuration mode.

EXT ADDR BUS, bits 4-0.

These five bits allow DMA accesses to extended memory; they select one of 32 maps.

*The AUTO bit is used for self-configured DMA only; it has no meaning when performing standard DMA.

Figure 2-1. General Bit Definitions for Control Word 1

Self-Configured DMA Operation

After receiving the STC in Register 20, the self-configuration control logic fetches the word addressed by the contents of Register 20 and loads this word into Register 21. Assume bit 7 of this word is set, signifying a quadruplet. The contents of Register 20 are incremented during the memory access. The new value of Register 20 is used as the address of the next memory read. This next word is loaded into the control register on the interface by means of a virtual OTA 31 generated by the IOP chip. Register 20 then is incremented for the next read and this new data is loaded into Register 22. Register 20 is again incremented. The new value of Register 20 is used to address the fourth word of the current quadruplet. This fourth word is loaded into Register 23. The value of Register 20 is again incremented, pointing to the beginning of the next triplet/quadruplet.

The DMA operation just loaded is then started as soon as the interface is ready. When the DMA operation terminates, if bit 11 of Register 21 is clear, an interrupt request is generated. If the DMA operation terminated due to either an end-of-transfer indication from the interface or a word/byte count transition from - to 0, then the DMA self-configuration logic writes the word-count residue into the location formerly occupied by the current DMA operation's word/byte count. Operation of the self-configuration feature is continued, as noted above, for the next triplet/quadruplet.

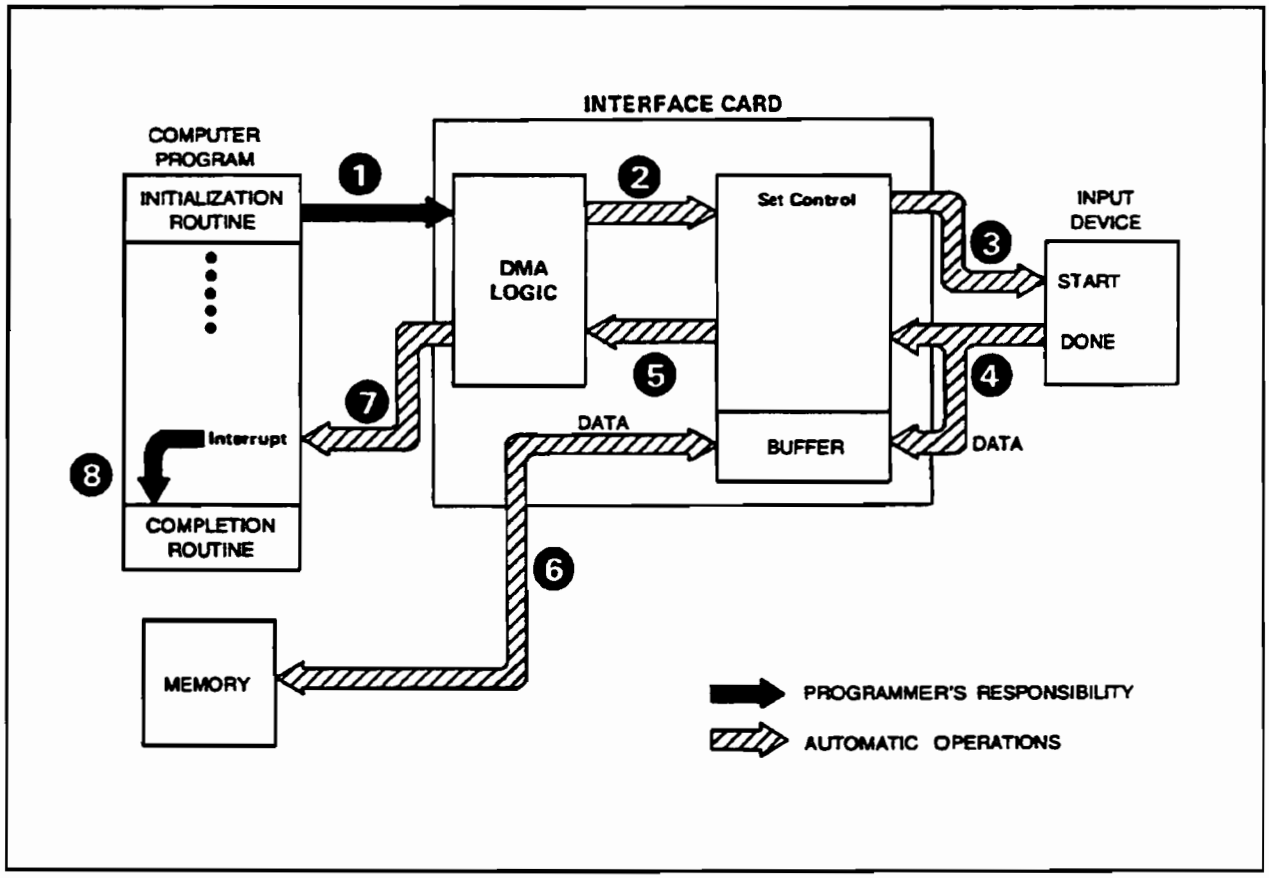
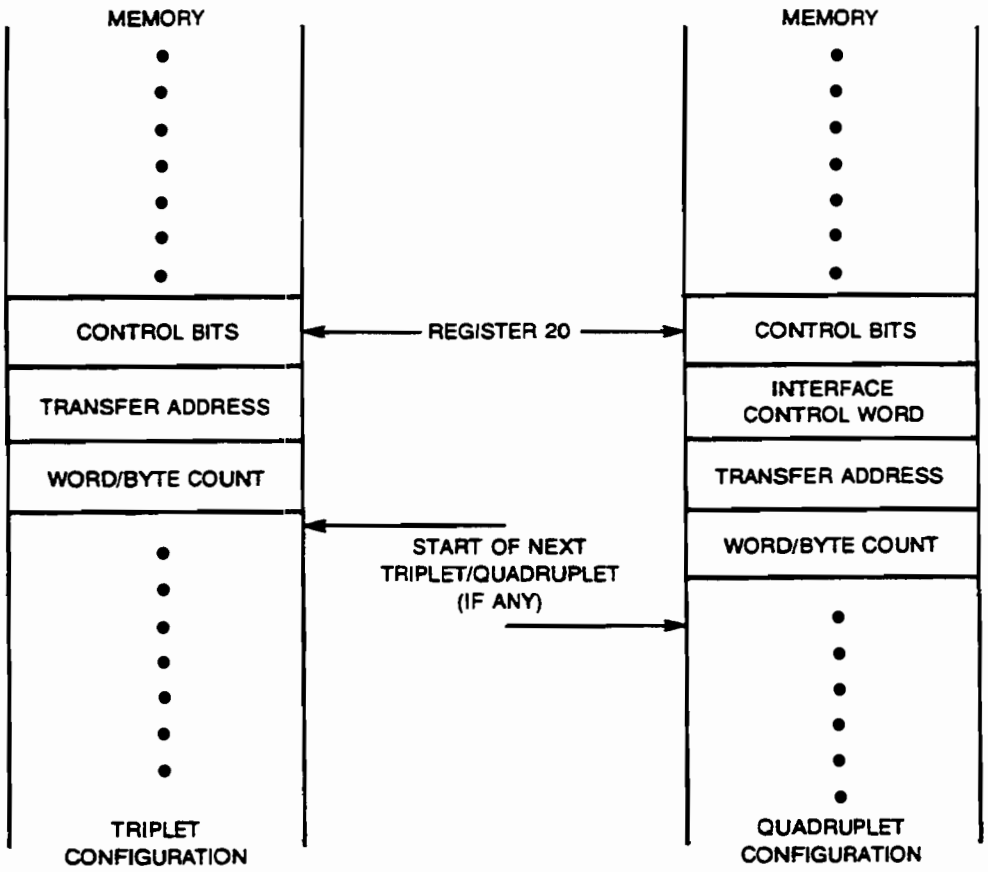


Figure 2-2. Standard DMA Input Transfer



REGISTER NUMBER 20 POINTS TO THE STARTING ADDRESS OF THE TRIPLET/QUADRUPLET.

THE "CONTROL BITS" ARE LOADED INTO DMA REGISTER NUMBER 21.

THE "TRANSFER ADDRESS" IS LOADED INTO DMA REGISTER NUMBER 22.

THE "WORD/BYTE COUNT" IS LOADED INTO DMA REGISTER NUMBER 23.

THE "INTERFACE CONTROL WORD" IS LOADED INTO REGISTER NUMBER 31.

8200-77

Figure 2-3. DMA Self-Configuration Feature

Self-Configured DMA Termination

The operation of the DMA self-configuration feature continues as described in the preceding paragraphs until one of the following events occurs:

a. A CLC to register number 20 is executed. This serves to inhibit the self-configuration logic from advancing its pointer to the next triplet/quadruplet, while still allowing the current DMA to continue. An STC to register number 20 allows the self-configuration feature to continue.

b. A CLC to register number 21 is executed. This stops the current DMA operation at its present state of operation.

c. A CLC to register number 22 is executed. This aborts the current DMA operation and causes the self-configuration logic to advance to the next triplet/quadruplet. No interrupt is generated by the aborted transfer.

d. A CLC to register number 23 is executed. This stops the operation of the self-configuration logic and aborts the transfer in progress.

e. The first word of a triplet/quadruplet is read with the CONT bit clear, indicating that there are no further DMA triplets/quadruplets. This sets the flag on register number 20, and will generate an interrupt request, at the termination of the current transfer.

Programmed I/O

The following paragraphs describe how data is transferred between memory and input/output devices without using DMA. This includes interrupt-driven I/O transfers and non-interrupt I/O transfers.

Interrupt-Driven I/O

By using the processor interrupt system, the programmer can be performing computation in parallel with I/O operations without having to periodically test the status of the device. The device will interrupt the program when it requires attention, suspending the program until it receives the required service.

Input Data Transfer (Interrupt Method)

Figure 2-4 illustrates an example of the sequence of events required to input data using the interrupt method. Note that some operations are under control of the computer program (programmer's responsibility), and some of the operations are automatic. Assume that the Global Register has been loaded and enabled and the I/O interface's control register has been loaded.

The operations begin (1) with the programmed instruction STC 30,C which sets the Control bit (Control 30) and clears the Flag bit (Flag 30) on the I/O interface. Since the next few operations

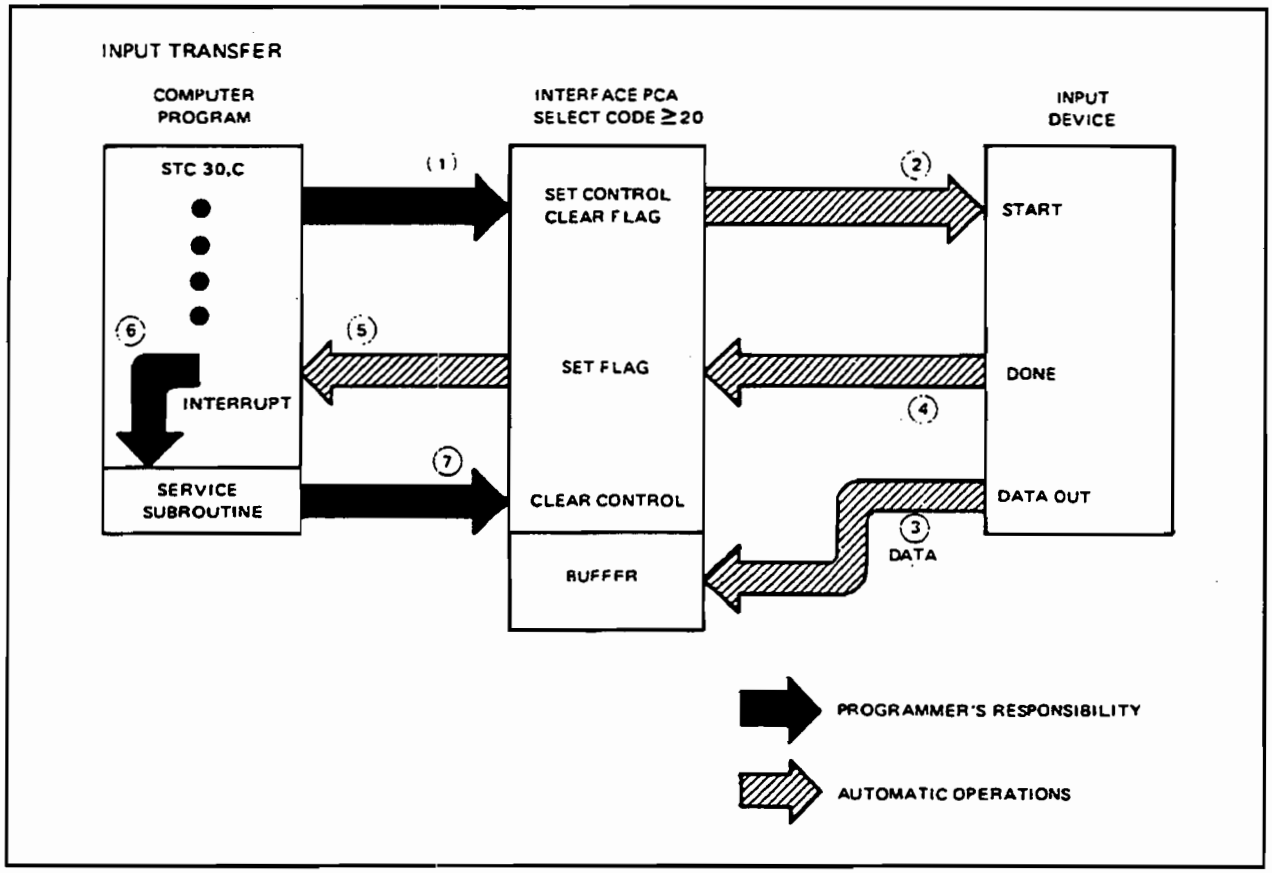


Figure 2-4. Input Data Transfer (Interrupt Method)

are under control of the hardware, the computer program may continue the execution of other instructions. Setting Control 30 causes the card to output a Start signal (2) to the device, which sends out a data character (3) and then asserts the Done signal (4).

The device Done signal sets the Flag bit, which in turn generates an interrupt (5) provided that the interrupt conditions are met; that is, the interrupt system must be on (STF 0 previously given) and Control 30 is set (done in step 1).

The interrupt causes the current computer program to be suspended and control is transferred to a service subroutine (6). It is the programmer's responsibility to provide the linkage between the interrupt location and the service routine by placing a JSB,I to the beginning of the service routine in the memory location corresponding to the I/O interface's select code. It is also the programmer's responsibility to include in the service subroutine the instructions for processing the data (for example, loading the data into an accumulator, manipulating it if necessary, and storing it into memory).

The service subroutine may then issue further STC 30,C instructions to transfer additional data items. One of the final instructions in the service subroutine after all transfers are done must be a CLC 30,C. This step (7) restores the interrupt capability to lower priority interfaces and returns the I/O interface to its static "reset" condition (Control clear and Flag clear). This condition is initially established by the computer at power turn-on, and it is the programmer's responsibility to return the I/O interface to the same condition on the completion of each data transfer operation.

At the end of the subroutine, control is returned to the interrupted program via previously established linkages .

Output Data Transfer (Interrupt Method)

Figure 2-5 illustrates an example of the sequence of events required to output data using the interrupt method. Again, note the distinctions between programmed and automatic operations. Assume that the Global Register has been loaded and enabled and that the I/O card's control register has been loaded. It is also assumed that the data to be transferred has been loaded into the A-register and is in a form suitable for output.

The output operation begins with a programmed instruction (OTA 30) to transfer the contents of the A-register to the I/O interface buffer (1). This is followed (2) by the instruction STC 30,C which sets the Control bit (Control 30) and clears the Flag bit (Flag 30) on the I/O interface. Since the next few operations are under control of the hardware, the computer program may continue the execution of other instructions. Setting the Control bit causes the interface to output the buffered data (3) followed by a Start signal (4) to the device, which writes (for example, records, stores, etc.) the data character and asserts the Done signal (5).

The device Done signal sets the interface's Flag 30, which in turn generates an interrupt (6) provided that the interrupt system is on, and Control 30 is set (done in Step 2). The interrupt causes the current computer program to be suspended and control is transferred to a service subroutine (7). It is the programmer's responsibility to provide the linkage between the interrupt location and the service subroutine by placing a JSB,I to the beginning of the service routine in the memory location corresponding to the I/O interface's select code. The detailed contents of the subroutine are also the programmer's responsibility and the contents will vary with the type of device.

The subroutine may then output further data to the I/O interface and reissue the STC 30,C command for additional data item transfers. One of the final instructions in the service subroutine must be a clear control (CLC 30,C). This step (8) allows lower priority devices to interrupt and restores the I/O interface to its static "reset" condition (Control clear and Flag clear). At the end of the subroutine, control is returned to the interrupted program via the previously established linkages.

Interrupt Processing

An interrupt request may be generated by many different events on an I/O interface card, including the following:

a. A device completion (non-DMA) or an interface exception condition (flag 30).

b. DMA completion (flag 20 or 21).

c. DMA Parity Error (flag 22).

The SFS and SFC instructions may be used to distinguish between these possibilities. The first flag to be tested should be the one most likely to cause an interrupt, or the one requiring the fastest processing. If the self-configuration feature were used, the code could look like:

```
WHICH   LDA SC
        OTA 2,C          set up Global Register
        SFC 23B          any DMA flags?
        JMP DMAI         DMA interrupt
        SFC 30B          device flag?
        JMP DVCI         device interrupt
        JMP ERROR
DMA1    SFC 22B          was it parity?
        JMP PARI
        SFC 20B          was it self-configuration completion?
        JMP CONI
        SFC 21B          transfer termination?
        JMP XFERI
ERROR                    <This card didn't request the interrupt.>
```

Programming Example (Interrupt Method)

In an operating system environment, one task may be executing during another's I/O wait time, the second task to be activated on completion of its I/O operation. The interrupt on I/O operation completion allows the operating system to perform this task activation.

In the following examples, an interrupt-driven scheduler is assumed.

Input Initiation

An input request by a task would activate the following code:

```
        LDA SCI
        OTA 2,C          set up Global Register
        LDA CW
        OTA 31B          set up interface
STARTI  STC 30B,C        start operation
        JMP CNTNU        wait for interrupt
```

The interface is configured with the proper control bits (contained in CW). Then the device is started. Control is transferred back to the RTE scheduler to await the completion interrupt. The control flip-flop of the interface card must be set and the interrupt system enabled before an interrupt may occur.

Input Operation

On receipt of the interrupt, the scheduler is assumed to save the current machine state, then route the interrupt to the appropriate driver:

```
C.XX    LDA SCI
        OTA 2,C          set up Global Register
        LIA 30B          get next data
        STA DATA,I       store it
        ISZ DATA         bump pointer
        ISZ COUNT        check if more needed
        JMP STARTI       go start next transfer
```

Once the data has been retrieved and stored, the requested input count is checked. If more data is needed, a new operation is initiated and control is again returned to the scheduler. If no new data is needed, control passes to the next section of code.

Input Termination

Assume that the operating system clears the interface flag (CLF 30B) on receipt of the device interrupt, all that is necessary to terminate the operation is to clear the interface control flip-flop, disabling any further interrupts:

```
        CLC 30B
        JMP DONE
```

It may be appropriate to interrogate the interface's status register before clearing it:

```
        LIA 32B
        STA STATS,I
        CLC 30B
        JMP DONE
```

Output Initiation

On output, the data must be placed into the interface output register before initiating the device operation, in case the device is capable of quickly taking the data. The code segment would be:

```
        LDA SCO
        OTA 2,C          set up Global Register
        LDA CW
        OTA 31B          set up interface
STARTO  LDA  DATA,I
        OTA 30B          output first data
        STC 30B,C        start operation
        JMP CNTNU        wait for interrupt
```

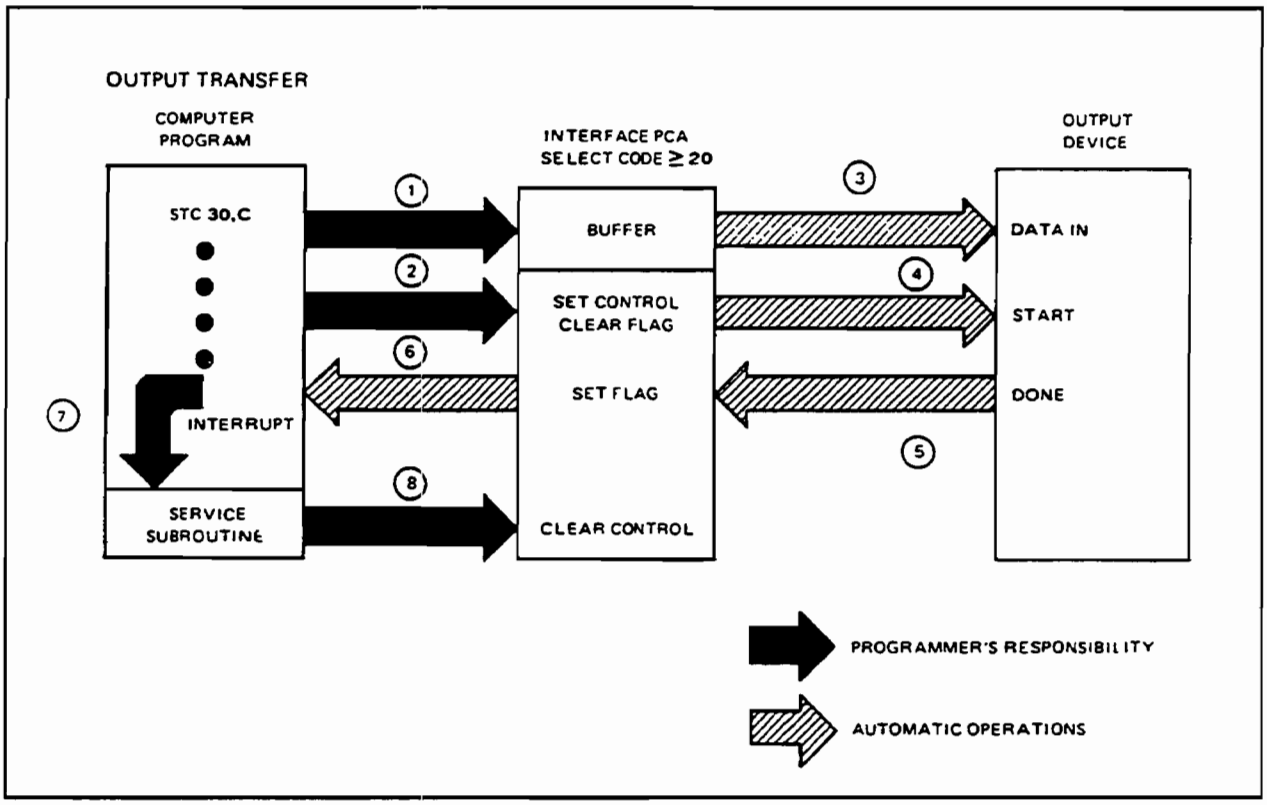


Figure 2-5. Output Data Transfer (Interrupt Method)

Output Operation

When each output is completed, an interrupt will be generated. Assume the operating system clears the interface flag (CLF 30B) before entering the driver. The code segment would be:

```
C.YY    LDA SCO
        OTA 2,C          set up Global Register
        ISZ DATA         bump pointer
        ISZ COUNT
        JMP STARTO       go start next transfer
```

Output Termination

To terminate the operation, the interface status could be read:

```
LIA 32B
STA STATS,I
```

Then, the interface cleared:

```
CLC 30B,C
JMP DONE
```

Non-Interrupt I/O

Non-interrupt I/O is conducted between the processor registers and the interface registers, with the status of the interface being determined by the use of the SFS (device ready) or SFC (device busy) instruction. The interrupt system is not used in this mode of transfer. The following code is only for illustrative purposes. Refer to the appropriate software manual for a complete diskussion on how to program an I/O interface.

Input Initiation

Assume that a 16-bit parallel device is connected to an interface card as an input-only device. The code to initiate the transfer would be as follows:

```
LDA SCI
OTA 2, C          set up Global Register
LDB CW
OTB 31B           configure interface
STC 30B,C         start device
```

Note that the instruction STC 30B,C serves two purposes: it clears the interface flag, initializing it for the subsequent test operation; and it issues a Device Command pulse that signals the device to commence sending data. The interface and device should be configured by an OTA/B 31B prior to issuing the start command, to ensure proper operation.

Input Operation

Once the device has been started, the program may do one of two things, either do some other processing (counting on having some amount of time before the device completes), or simply waiting for the device to complete (as when the next operation depends on the completion of the input that was just initiated). In either case, the interface status will be checked for completion with one of the following sequences:

```
LDB SCI
OTB 2,C           set up Global Register
SFC 30B
JMP GETIT         it's ready with data
<continue processing>
```

or

```
LDB SCI
OTB 2,C           set up Global Register
SFS 30B           Test—
JMP *-1           —wait if not done
<process this input> data is now ready
```

To read the data:

```
LDB SCI
OTB 2,C          set up Global Register
LIA 30B          get data word
```

To start the next operation:

```
LDB SCI
OTB 2,C
STC 30B,C
```

Note that each of the above sequences shows the Global Register being configured, but this is not always necessary. If the Global Register already contains the select code of the interface, then it need not be reconfigured. For the case where the interface must wait for the input, the following sequence could be used:

```
LDB SCI
OTB 2,C          set up Global Register
LDA CW
OTA 31B          set up interface
STC 30B,C
SFS 30B
JMP *-1
LIA 30B          get next data
<continue processing>
```

Input Termination

Once the last data word has been read, the interface may be reset to a known state by the following:

```
LDB SCI
OTB 2,C          set up Global Register
CLC 30,C         turn off interface
```

No Device Command pulse is issued, so the device will not initiate another operation.

Output Initiation

Again assume a 16-bit parallel device connected to an I/O interface card, but this time as an output-only device. The code to initiate the transfer could be:

```
LDA SCO
OTA 2,C          set up Global Register
LDA CW
OTA 31B          set up interface
LDA DATA,I       get 1st data
```

```
OTA 30B          output to data register
STC 30B,C        start the output
```

The Global Register is configured and enabled, the interface is configured, the data is put into the output register, and the Device Command pulse is issued.

Output Operation

Once the output has been initiated, the program is free to perform other tasks, subject only to any time constraints imposed by the device itself (that is, possible under-run when transferring to a disk). If no further transfers are to be performed, the interface could be left to complete whenever it will, with the completion going unnoticed, or the completion could be tested for and the interface cleared (the latter is the preferred method).

To test for operation completion, the following code could be used:

```
LDA SC0
OTA 2,C          set up Global Register
SFS 30B          test
JMP *-1          not done yet
```

To initiate the next operation, proceed in a similar manner as before:

```
ISZ DATA         bump data pointer
LDA DATA,I
OTA 30B          output data
STC 30B,C        start device
```

Output Termination

It is advisable to wait for the last transfer to complete before clearing the interface. If it is cleared immediately after the last transfer is initiated, when the last transfer is completed, it will set the flag, putting the interface into non-initialized condition.

To clear the interface:

```
LDA SC0
OTA 2,C
CLC 30B,C
```

3

HP Interfaces

This chapter contains a description of the available HP A-Series I/O cards; these cards can be used to solve a wide variety of interfacing problems. Also listed are the manuals that describe the I/O cards. If a custom interface is required, then the HP 12010A Breadboard Interface card, with the I/O Master, is available to simplify design.

I/O Interface

Each I/O interface card is comprised of two sections, an I/O Master and the peripheral device interface section.

The I/O Master provides a standard interface to the backplane which is used by all interface cards. The I/O Master includes the I/O processor (IOP) chip and buffers for control, data, and addresses. The IOP chip contains the logic for all interactions with the backplane, central processor, and memory. The I/O functions include:

a. Recognition and execution of I/O instructions.

b. Direct Memory Accessing (DMA).

c. Virtual Control Panel (VCP) mode processing.

d. Vector address generation for interrupts.

e. Interrupt request priority conflict resolution.

f. Pending interrupt signal generation for the central processor.

The peripheral device interface performs all of the necessary functions to interface the external device with the I/O Master and the computer. The peripheral device section is different for each type of I/O interface card. The interfacing signals could include start and stop flags, control and status bits, and data input and output registers.

Available Interfaces

The following general-purpose interface products are available from Hewlett-Packard. Additional information about these interfaces are available from any HP Sales and Service Office. The basic supporting manual for each interface is listed in Table 3-1, and the I/O Master circuitry of each interface card is explained in Chapter 4 of this interfacing guide.

1. HP 12005A/B Asynchronous Serial Interface.
2. HP 12006A Parallel Interface.
3. HP 12007B HDLC Modem Interface.
4. HP 12008A PROM Storage Module.
5. HP 12009A HP-IB Interface.
6. HP 12016A SCSI Host Bus Adapter Card.
7. HP 12040D 8-Channel Asynchronous Multiplexer Subsystem.
8. HP 12044A HDLC Direct Connect Interface.
9. HP 12076A LAN/1000 Link Interface.

HP 12010A Breadboard Interface

The HP 12010A Breadboard interface provides the standard backplane interface logic (I/O Master) along with space for 60 16-pin wire-wrap sockets for user-designed custom interfaces. The printed circuit layout is based on a 0.100 by 0.300 inch matrix which accommodates any mix of dual or single in-line ICs. All signals needed by a designer are brought to convenient wire-wrap posts at the location where the backplane interface logic ends and the user area begins.

The 12010A is designed to give the user the ability to quickly design and implement custom interfaces to the A-Series computer. Complete documentation is included to guide the interface design process. A detailed description of the breadboard interface is contained in Chapter 4.

Features

- Standardized interface to A-Series backplane.
- Built-in DMA capability for optimum I/O efficiency.
- Substantially lowers the engineering investment required to design I/O cards.
- Accomodates up to 6016-pin wire-wrap sockets or any other combination of dual in-line integrated circuit sockets with different numbers of pins.
- All signals are TTL compatible.

Table 3-1. Manuals for A-Series I/O Cards

TITLE	HP PART Number
HP 12005A/B Asynchronous Serial Interface Reference Manual	12005-90001
HP 12006A Parallel Interface Reference Manual	12006-90001
HP 12007B HDLC Modem Interface Installation and Service Manual	5955-7694
HP 12008A PROM Storage Module Reference Manual	12008-90001
HP 12009A HP-IB Interface Reference Manual	12009-90001
HP 12016A SCSI Host Bus Adapter Card Installation and Reference Manual	12016-90001
HP 12040D 8-Channel Asynchronous Multiplexer Subsystem Installation and Reference Manual	12040-90123
HP 12044A HDLC Direct Connect Interface Installation and Service Manual	5955-7694
HP 12076A LAN/1000 Link Installation Manual	12076-90001

4

I/O Master

This chapter contains the theory of operation of the I/O Master, with emphasis placed on the user interface rather than the backplane interface. The theory is divided into two parts, a functional description of the simplified I/O Master block diagram and the IOP chip block diagram (Figures 4-1 and 4-2), followed by a circuit description.

Functional Description

The I/O Master manages all the I/O processing operations in the A-Series computer. Its capabilities include I/O instruction recognition and execution, direct memory accessing, and slave mode processing. The I/O Master consists of a 64-pin LSI I/O processor (IOP) chip, some memory request logic which is placed outside the chip for high-speed operation, and data, address, and chip bus buffers. The I/O chip and its support logic make up the I/O Master.

As the I/O processing manager, the I/O Master interfaces the device specific circuits to and from the backplane. The central processor and memory are essentially invisible to the device. To accomplish this manager function, the I/O Master decodes and executes all I/O instructions and selects the desired access to either the CPU or memory. It is capable of initiating, processing, and terminating DMA transfers, or accessing the CPU A-, B-, and P-Registers, or requesting an interrupt.

I/O Processor (IOP) Chip

By far, the most complicated circuitry on the I/O Master is internal to the IOP chip. The I/O Master circuitry external to the IOP serves the purpose of buffering signals into and out of the IOP, and performing several logic functions which, due to their speed-critical nature, had to be placed outside of the IOP. Figure 4-1 is a simplified block diagram of the entire I/O Master, showing the IOP chip and its support logic. Each of these blocks are described following this IOP description (see Figure 4-2). The chip bus is a 16-bit bidirectional bus over which all data is sent into and out of the IOP. The IOP must receive and drive the data bus and drive the address bus. For example, each instruction fetched by the CPU is sent into the IOP via the chip bus; during DMA transfers, the transfer address is read out via the chip bus. The IOP contains a register file whose registers are used for address recognition, identifying the interface by placing its address on the backplane, interrupt masking (permits the programmer to selectively inhibit interrupts), and to control DMA.

The instruction executor is a Custom Logic Array (CLA) controlled state machine designed specifically to interact with the A-Series CPU in the decoding and execution of I/O instructions. The CLA examines the instruction latch in the register file. When it recognizes an instruction that requires interaction with the CPU, it outputs a control word telling the CPU to increment its P-Register (for a SFS or SFC), or to prepare for a data transfer (OTA/B, LIA/B, and MIA/B).

The interrupt logic responds to the condition of its flag and control flip-flops (FLG 30 and CNTL 30). Whenever both are set and interrupts are not masked, the IOP chip requests an interrupt. Both FLG 30 and CNTL 30 can be set and cleared programmatically. In addition, FLG 30 can be set by hardware.

The DMA logic is used to enable DMA transfers and to generate DMA completion or parity interrupts. Either of two control flip-flops can be used to enable DMA, CNTL 20 or 21 (CNTL 20 for self-configured DMA). Associated flag flip-flops (FLG 20 or FLG 21), when set, generate DMA completion interrupt signals. An additional flag, FLG 22, is used for a DMA parity error interrupt.

The slave logic responds to a slave mode request (VCP request) from a device/interface, and directs the I/O Master to set up a slave mode cycle with the CPU.

Timing and Control Logic

To maintain uniform backplane loading, all timing and control signals are buffered by the I/O Master before being sent to the user.

Bus Control Logic

The key element of this block is a decoder controlled by the IOP chip. The decoder directs the flow of data between the backplane data bus and the user or the IOP. Two of eight states control data flow into or out of the I/O Master, leaving six states for control of the user's registers.

Bus Drivers and Receivers

This block contains a select code driver, a data bus driver, a data bus receiver/latch, and an address bus driver. On command from the bus control logic, these devices latch data from the data bus or drive data onto the data or address bus. At power up, the select code driver enables the interface's select code onto the chip bus so that the interface can "learn" its select code.

This chapter contains the theory of operation of the I/O Master, with emphasis placed on the user interface rather than the backplane interface. The theory is divided into two parts, a functional description of the simplified I/O Master block diagram and the IOP (I/O processor chip) block diagram (Figures 4-1 and 4-2), followed by a circuit description.

Slave and Handshake Logic

Two flip-flops are used to request a slave cycle (for example, as used by the Virtual Control Panel), and when the request is acknowledged by the CPU, to request a series of handshake cycles

required to complete the slave sequence. The CPU responds with a signal to complete each handshake. This same handshake logic/CPU exchange is carried out for each I/O instruction requiring communication with the CPU.

Interrupt Logic

The interrupt priority chain input goes to the IOP chip, and when combined with a priority disable signal from the IOP becomes the interrupt chain output. The IOP asserts an interrupt request and the CPU asserts an interrupt acknowledge signal to the request.

Memory Request Logic

The I/O Master asserts a memory request signal to "tie up" the backplane buses and a memory go signal to initiate a memory operation. The memory request is accompanied by two memory priority chain signals, to inhibit a memory "go" signal from lower priority requesters. Similarly, a higher-priority memory request inhibits a request from this I/O Master. If the backplane was congested with DMA operations, the central processor could be held off indefinitely; however, the signal CPUTURN (described in Chapter 2) is used in the A700, A900, and A990 computers to give the central processor the ability to periodically gain access to the backplane for one memory cycle.

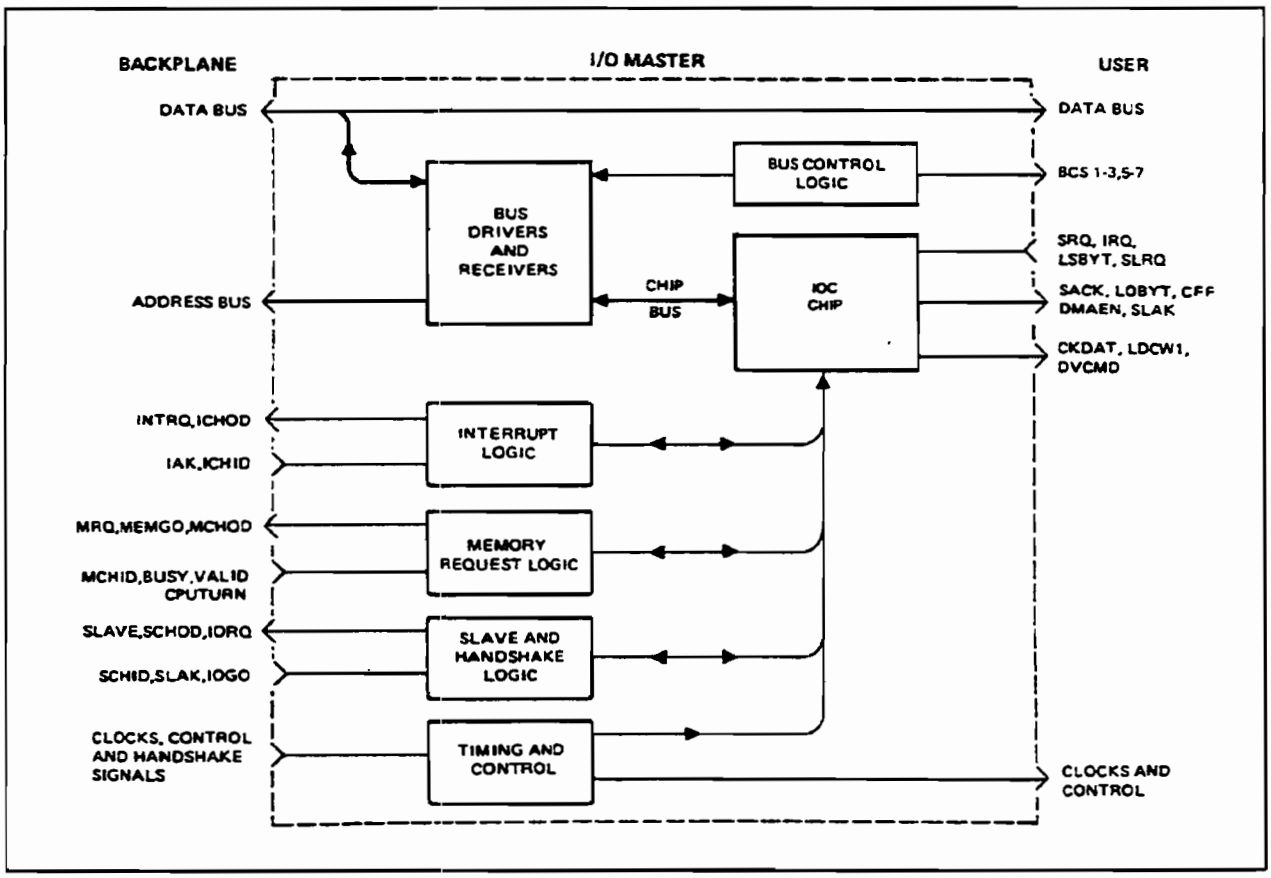
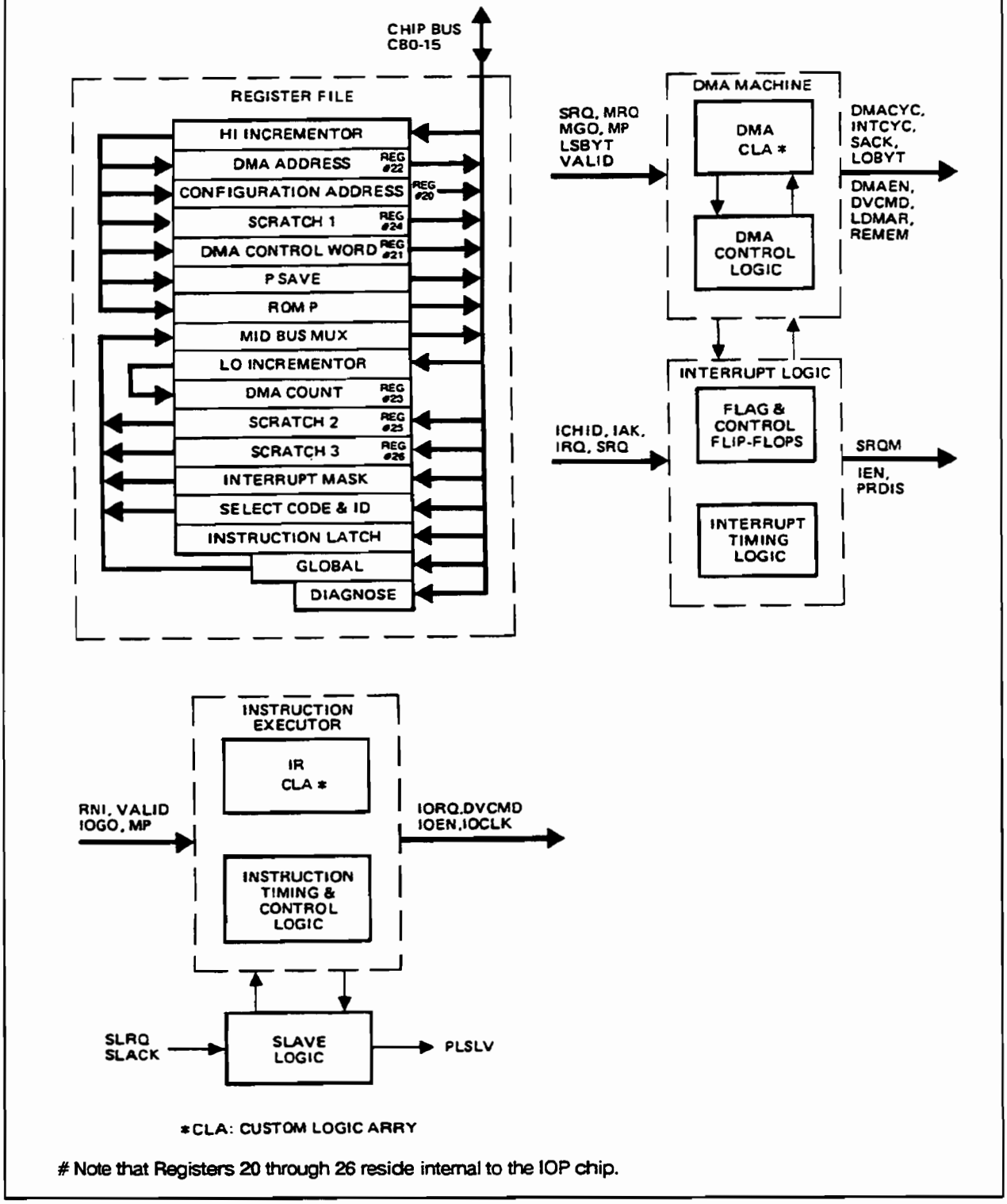


Figure 4-1. I/O Master Simplified Block Diagram



Note that Registers 20 through 26 reside internal to the IOP chip.

Figure 4-2. I/O Processor Chip Block Diagram

Table 4-1. Signal Names and Mnemonics

Mnemonic	Name	Origin
AB0–AB14	Address Bus 0 to 14	Backplane (P1–19–33)
BCS0–7	Bus Control State 0 to 7	I/O Master (U46)
BCW0–2	Bus Control Word 0–2	IOP
BPON	Buffered Power On	Backplane (P2–26)
BUSY	Memory Busy	Backplane (P2–20)
CB0–CB15	Chip Bus	IOP
CCLK	Communications Clock	Backplane (P2–22)
CFF	Control Flip–Flop	IOP
CHSRQ	Chip Service Request	I/O Master (U66–13)
CKDAT	Clock Data	I/O Master (U117–10)
CPUTURN	Central Processor Turn	Backplane (P2–1)
CRS	Control Reset	Backplane (P1–25)
CW1	Control Word 1	IOP
DB0–DB15	Data Bus 0 to 15	Backplane (P1–35–50)
DIAG	Diagnose Mode 7	IOP
DMACYC	DMA Cycle	IOP
DMAEN	DMA Enable	IOP
DVCMD	Device Command	IOP
FETCH	Fetch	I/O Master (U106–8)
IAK	Interrupt Acknowledge	Backplane (P2–13)
ICHID	Interrupt Chain In Disable	Backplane (P1–1)
ICHOD	Interrupt Chain Out Disable	Backplane (P1–2)
IEN	Interrupt Enable	IOP
INTCYC	Interrupt Cycle	IOP
INTRQ	Interrupt Request	Backplane (P2–6)
IOCLK	I/O Clock	IOP
IOEN	I/O Enable	IOP
IOGO	I/O Handshake Request Acknowledge	Backplane (P2–14)
IORQ	I/O Handshake Request	Backplane (P2–5)
IRQ	Interrupt Request	User Interface
LDCW1	Load Control Word 1	I/O Master (U107–6)
LDMAR	Load Memory Address Register	IOP
LOBYT	Lower Byte	IOP
LSBYT	Last Byte	User Interface
MCHID	Memory Chain In Disable	Backplane (P1–3)
MCHOD	Memory Chain Out Disable	Backplane (P1–4)
MCHODOC	Memory Chain Out Disable Open Collector	Backplane (P1–6)
MEMGO	Memory Go	IOP
MP	Memory Protect	Backplane (P2–7)
MRQ	Memory Request	IOP
PE	Parity Error	Backplane (P2–10)
PLSLV	Pull Slave	IOP
PON	Power On	I/O Master (U108–7–14)
PRDIS	Priority Disable	IOP
PULIOR	Pull I/O Handshake Request	IOP
REMEM	Remote Memory	Backplane (P–3)
REMOTE	Remote	IOP
RNI	Read Next Instruction	Backplane (P2–8)
RST	Reset	I/O Master (U117–13)
SACK	Service Request Acknowledge	IOP
SCEN	Select Code Enable	IOP
SCHID	Slave Chain In Disable	Backplane (P2–11)
SCHOD	Slave Chain Out Disable	Backplane (P2–12)
SCLK	Slow Clock	Backplane (P2–44)
SC0–4	Address Extension Bus	Backplane (P1–9–12, 17)

Table 4-1. Signal Names and Mnemonics (Continued)

Mnemonic	Name	Origin
SC5	Self Configure	Backplane (P1-18)
SLACK	Slave Acknowledge	I/O Master (U56-9)
SLAK	Slave Request Ackknowledge	I/O Master (U116-9)
Slave	Slave Request	I/O Master (U56-5)
SLRQ	Slave Request	User Interface
SRQ	Service Request	User Interface
SRQM	Service Request Mask	I/O Master (U106-6)
VALID	Data Valid	Backplane (P2-4)
WE	Write Enable	I/O Master (U37-19)

Note: The signals shown on the IOP chip (Figure 4-18) with a B prefix are buffered by U108 and U116 and are described in the above table without the prefix.

I/O Master Circuit Description

The following paragraphs provide a circuit description of the I/O Master. The I/O Master block diagram, schematic diagram, and IC pack diagrams located in this chapter may be helpful in following the circuit description.

Signal Names and Mnemonics

Table 4-1 lists all of the signal names and mnemonics used on the I/O Master. The origin column in the table refers to the signal origin relative to the I/O Master (to aid in locating it) and may not be the signal source.

Timing and Control Signals

The thirteen timing and control signals shown in Figure 4-3 are received from the backplane and are buffered and inverted in sections of tri-state buffers and inverters U108 and U116. Note that BPON is inverted twice so that both PON+ (active high) and PON- (active low) signals are available to the I/O Master and the user interface. Two of the signals shown below are generated on the I/O Master specifically for use by the user (FETCH- and RST-) and are explained below. The remainder of the signals are discussed in the following paragraphs.

FETCH–

The CPU asserts RNI– to signal an I/O instruction fetch. RNI holds until VALID– is received from memory, indicating the completion of the memory cycle. The signal FETCH–, generated during the coincidence of RNI and VALID, may be used to clock instructions into a user instruction register. The trailing (positive going) edge of FETCH should be used to clock or close the latch.

RST–

The assertion of CRS–, caused by the execution of CLC 0 instruction, or the non-assertion of BPON+ causes the assertion of RST–. The RST–, reset, signal may be used to reset all interface card logic to a known state.

Bus Control Logic

BCS0– through BCS7–. The 3 to 8 decoder (U46) outputs control the flow of data between the data bus and logic in the I/O Master and the user circuitry (Figure 4-4). The 3-bit bus control word (BCS0, 1,2) produced by the IOP chip determine which of the eight BCS lines will be asserted when U46 is enabled. U46 is enabled by IOEN– from the IOP or by the MEMGO FF (U86B). MEMGO– is enabled any time the I/O Master initiates a memory cycle. IOEN– is asserted by the IOP any time the I/O Master wishes to read from or write to the data bus, except during a DMA read. The assertion of MEMGO and then IOEN are used to keep the decoder outputs enabled for the required period of time. The bus control states are described in Table 4-2. Except for BCS 0 and 4, the bus control states are user definable. The functions described in Table 4-5 are for typical operations. Also refer to the BCS timing diagrams and specifications in Table 4-6 and Table 4-8.

Bus Drivers and Receivers

These devices control data flow into, out of, and within the I/O Master.

Data Bus Receiver/Latch

The octal transparent D-type latches U57 and U58 (Figure 4-5) accept data during the assertion of VALID+ or IOGO+, capturing the contents of the backplane data bus on the trailing edge (deassertion) of these signals. Their outputs are enabled onto the IOP chip bus during the assertion of BCS0– or BCS2–. Thus, if BCS0– is asserted at the same time VALID is asserted, any data appearing on the backplane data bus will pass through the "transparent" latch and appear on the IOP chip bus. The latch function is required in order to hold data for the IOP after the backplane data bus has become invalid.

Data Bus Driver

The assertion of BCS4– enables the two octal non-inverting line drivers U47 and U48 (Figure 4-6), thus enabling the IOP chip bus onto the backplane data bus. This occurs, for example, during the execution of an LIA instruction which reads the contents of a register internal to the IOP.

Select Code and Board ID Driver

Switches U1S3–S8 are used to set the board select code. The breadboard ID code is preset at the factory, the select code switches must be set by the user. The two non-inverting line drivers U27 and U28 (Figure 4-7) transmit these codes to the IOP chip bus. At power on before BPON+ is asserted SCEN– is asserted, thus enabling the 6-bit select code and 10-bit identifier word onto the IOP bus. Upon assertion of BPON+ the IOP latches this word into an internal register and then deasserts SCEN–. The select code, as described previously, is used as an I/O address for I/O instructions. The board ID (identifier) is assigned by board type and used by diagnostic programs.

Address Drivers

To initiate a memory operation, the IOP chip places a memory address on the chip bus and asserts the signal LDMAR+ (Load Memory Address Register) to load the address into the octal transparent latches U37 and U38 (Figure 4-8). The deassertion of LDMAR latches the address into the address register, thus permitting the chip bus to be used to transfer data during the upcoming memory operation. When the memory request logic is set (described in the following paragraphs) the "not" Q output of the MEMGO FF U86B enables the address onto the backplane address bus.

To permit the memory system to distinguish DMA self-configuration from DMA data transfers, the signal SELFC– is asserted only during DMA self-configuration. The SELFC– driver, NAND gate U96A, is enabled by the Q output of the MEMGO FF and: BCS0, asserted while fetching the DMA control word; BCS2, asserted while fetching the user control word; and BCS4, asserted when DMA writes its word count residue into memory.

The address driver U18 and its associated circuitry are used to drive the 5-bit Address Extension Bus onto the backplane. These 5 bits are loaded into U18 with the assertion of CW1– and IOCLK+ during standard DMA initialization or with the assertion of CW1– and VALID+ during self-configured DMA initialization. The Address Extension Bus is driven onto the backplane simultaneously with AB0 – AB14.

The signal REMEM– is asserted to indicate that the simultaneous MEMGO– which occurs should initiate a memory cycle with the remote memory. Any memory card in the system should ignore MEMGO– if it occurs with REMEM–. The remote memory feature is not presently implemented by HP.

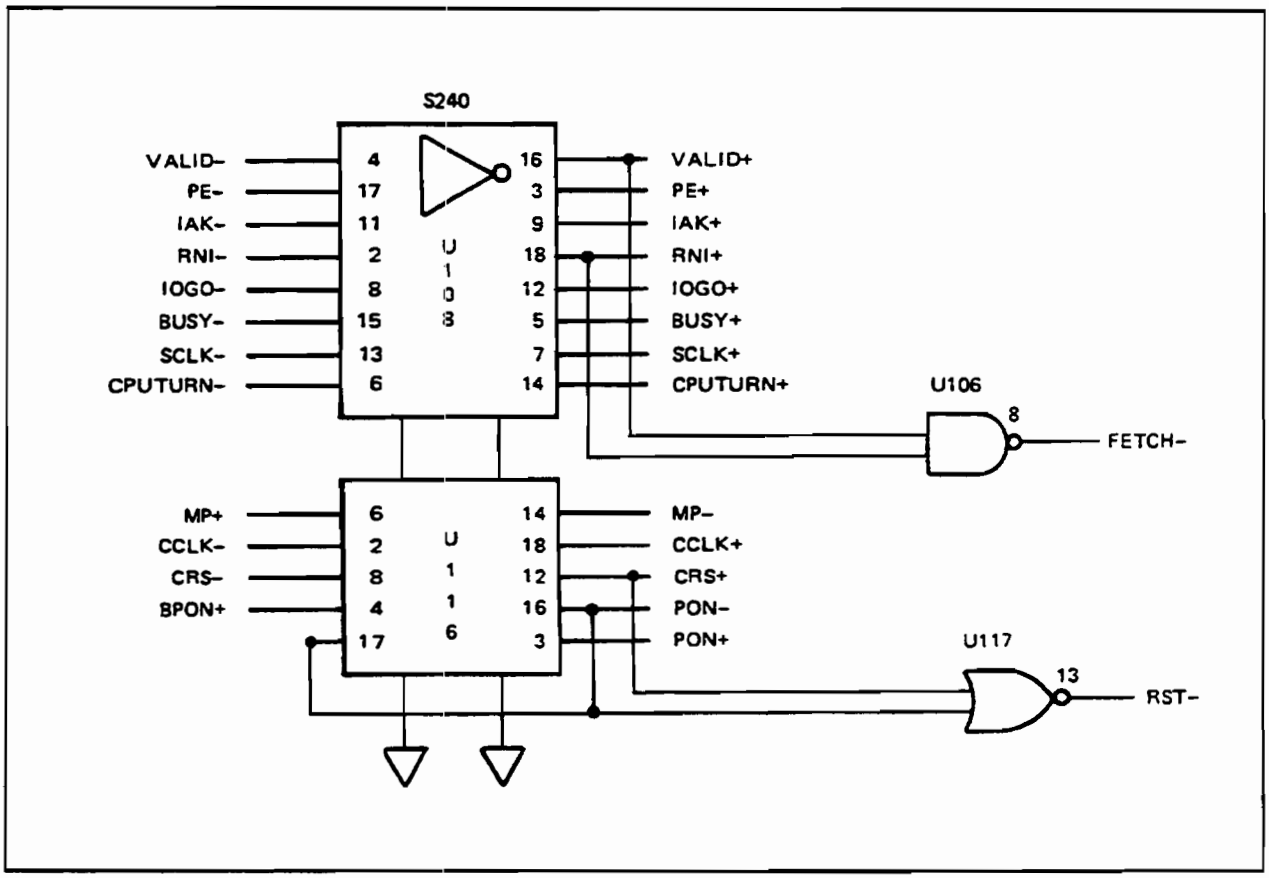


Figure 4-3. Timing and Control Signals

Memory Request Logic

DMA Memory Requests

The IOP chip initiates a DMA memory request by asserting DMACYC- to set the MRQ FF (U86A, Figure 4-9). With DMACYC asserted, and IAK+, CPUTURN+, and IOGO+ all deasserted, the MRQ FF sets. This causes the assertion of MRQ- (U118C), MCHODOC(U118B), and MCHOD- (U36A).

The assertion of MRQ inhibits any operation by the CPU which could require use of the backplane buses (except for an A600 computer). If an A600 CPU requests a memory access during the same cycle, MRQ is asserted by an interface card and the CPU is granted the next memory cycle. The priority line MCHOD "daisy chains" to MCHID- on the next lower priority card and inhibits the MEMGO FF. This prevents all lower priority DMA requests.

In systems using the 16- or 20-slot card cage, the delay time for MCHOD to ripple through the priority chain would be excessive, therefore, the signal MCHODOC- is used to perform the inhibit function for I/O cards in slots 9 through 16 (16-slot card cage) or slots 11 through 20 (20-slot card cage). MCHODOC is not used in the 6-slot card cage.

Table 4-2. Bus Control States

BCS	Function
BCS = 0	Enables the backplane data onto the IOC bus.
BCS = 1	Enables the backplane data bus onto the user data bus during DATA output operations. BCS1- is asserted with an OTA 30 or on DMA output.
BCS = 2	Enables the backplane data bus onto the user data bus during CONTROL WORD output operations. BCS2- is asserted with an OTA 31 or during DMA self-configuration.
BCS = 3	User definable.
BCS = 4	Enables the IOC bus onto the backplane data bus.
BCS = 5	Enables the user data bus onto the backplane bus during DATA input operations. BCS5- is asserted with an LIA 30 or DMA input.
BCS = 6	Enables the user data bus onto the backplane data bus during CONTROL WORD input operations. BCS6 is asserted with an LIA 31.
BCS = 7	User definable.

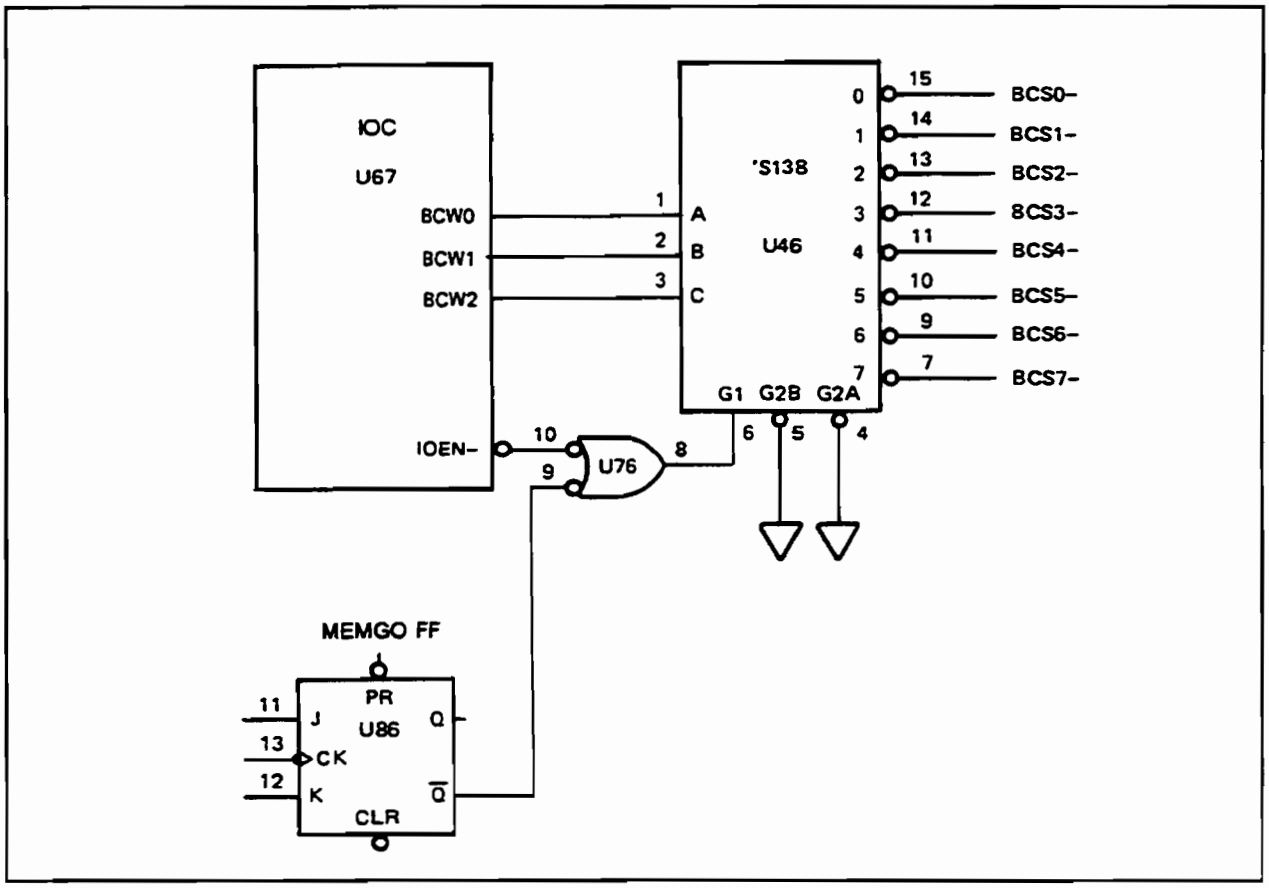


Figure 4-4. Bus Control Logic

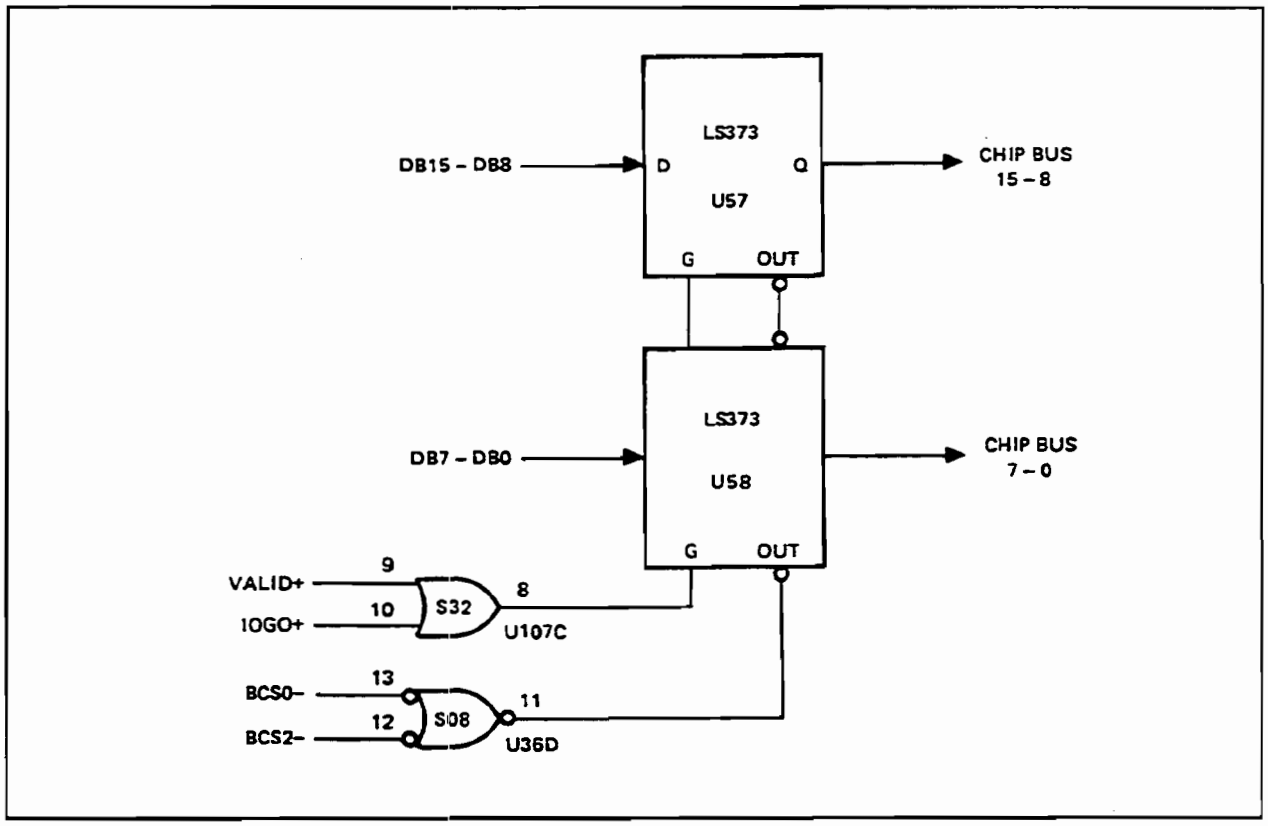


Figure 4-5. Data Bus Receiver/Latch

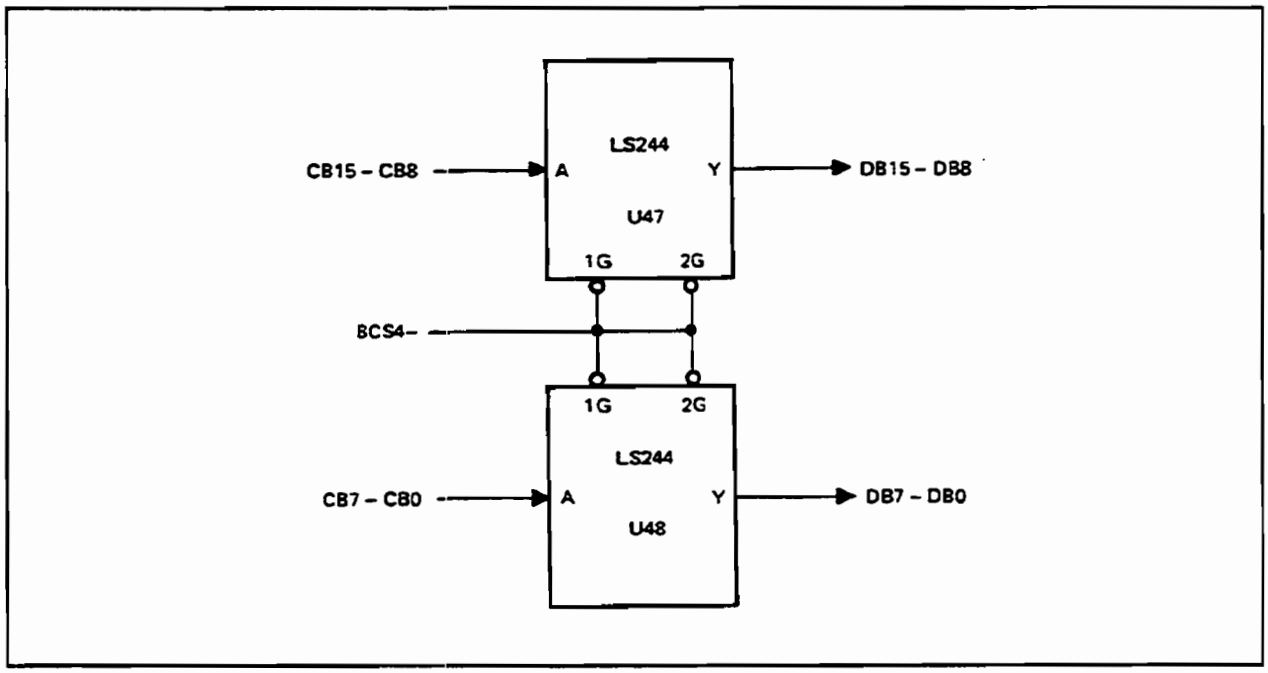


Figure 4-6. Data Bus Driver

Memory Cycle Initiation

The IOP chip asserts INTCYC+ to start a memory cycle. If the memory is not busy (BUSY+ U66C) and a higher priority device is not requesting (MCHID- U76B), the MEMGO FF (U86B) is set, asserting MEMGO. Also, the memory address is driven onto the backplane as described in this chapter. MEMGO is released after being held low for one cycle of SCLK.

For DMA memory operations, the IOP chip asserts DMACYC-, waits for MRQ+, and then asserts INTCYC+ to initiate the memory cycle.

During assertion of IAK+ (the interrupt acknowledge sequence), MRQ+ is not required. The IOP chip need only assert INTCYC+ to start the memory operation.

Memory Handshake Timing

Memory handshake timing is shown in Figure 4-10.

Slave and Handshake Logic

Slave Sequence

The slave mode capability must be enabled at power on by having switch U1S1 on the user interface in the VCP position (closed) when PON is asserted. If this condition is not met, the IOP chip will not recognize any subsequent changes of the SLRQ+ (slave request) signal from the user interface. Note that at most one card may have slave mode capability at power on. A computer self-test error message will be displayed on the processor LEDs (frontplane LEDs on A700) if the VCP switch is enabled on more than one interface. Refer to the applicable computer reference manual listed in Table 1-2 for VCP program considerations.

In response to the assertion of SLRQ+ from the user interface or in response to a HLT instruction, the IOP chip asserts PLSLV+, setting the SLAVE FF (U56A, Figure 4-11). The Q output of U56A asserts SLAVE- (U96C) to request a slave handshake cycle from the processor. The processor acknowledges the slave request by raising SCHOD (deasserts) which daisy chains down the SCHID/SCHOD lines from card to card until it reaches the requester. On the slave requester, the SLACK FF (56B) is set by the combination of SCHID and the Q output of the SLAVE FF to the CLR input of U56B, thus asserting SLACK+ to the IOP and IORQ- (U76D and U96B) to the processor. The IOP chip responds to SLACK+, the slave acknowledge signal, by asserting PULIOR-, which holds IORQ- asserted after the SLACK FF is reset. The assertion of IORQ- requests an I/O handshake with the processor. Because the delay from SLACK through the IOP chip to assert PULIOR- is too long to meet the processor timing requirements, it is necessary to initially cause the assertion of IORQ- directly from the SLACK FF.

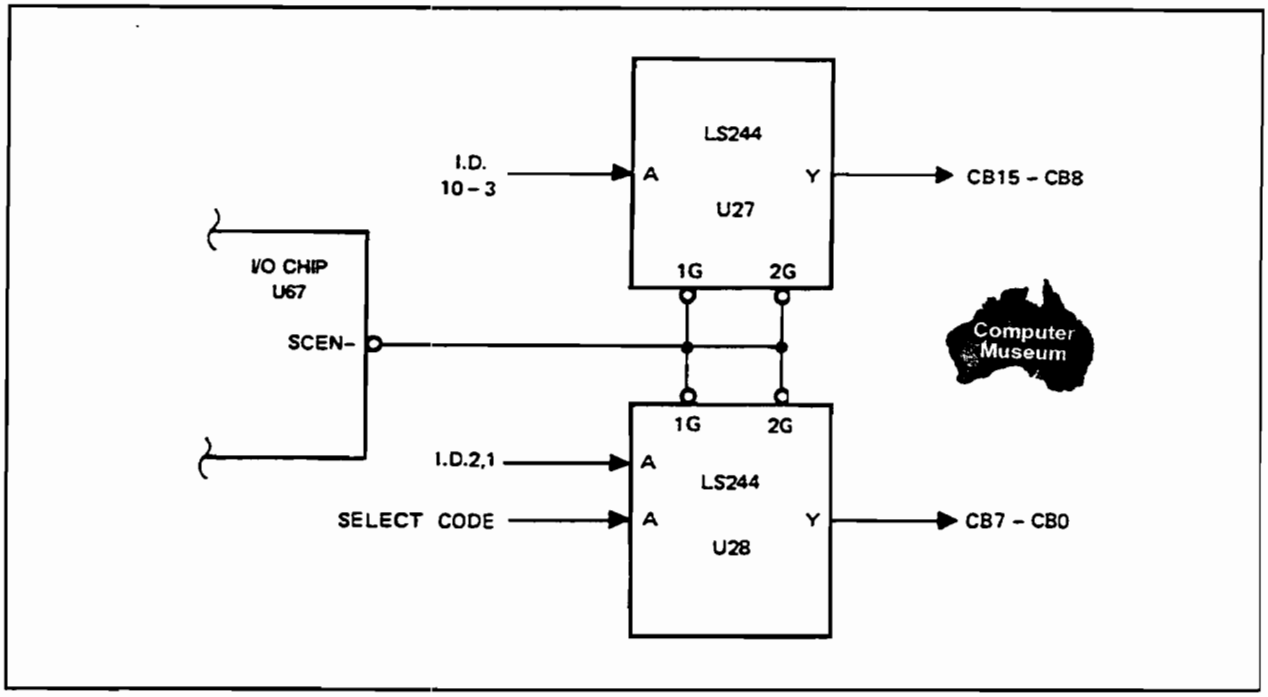


Figure 4-7. Select Code and ID Driver

SLACK+ also causes the I/O chip to deassert PLSLV+, thus clearing the SLAVE FF (on the next rising edge of SCLK-) which in turn clears the SLACK FF. The assertion of PULIOR- occurs before the SLACK FF is reset, giving an uninterrupted assertion of IORQ.

The SLAVE FF and SLACK FF are clocked on the rising (positive going) edge of SCLK-.

Slave Mode Timing

Slave mode timing is shown in Figure 4-12.

I/O Instruction Handshake Logic

During an I/O instruction handshake, the assertion of IORQ is always controlled by the IOP chip's assertion of PULIOR.

I/O Handshake Timing

I/O Handshake timing is shown in Figure 4-13.

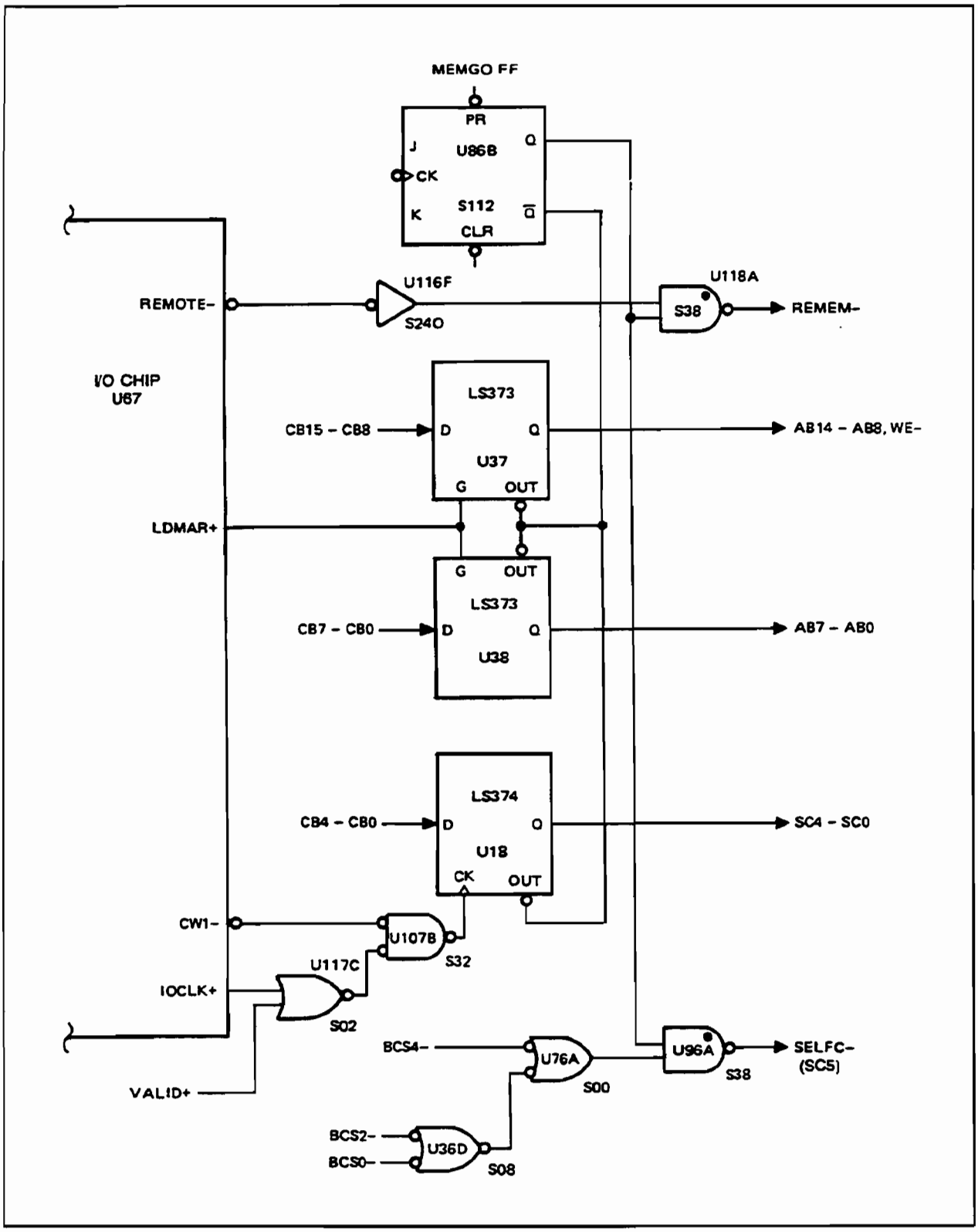


Figure 4-8. Address Drivers

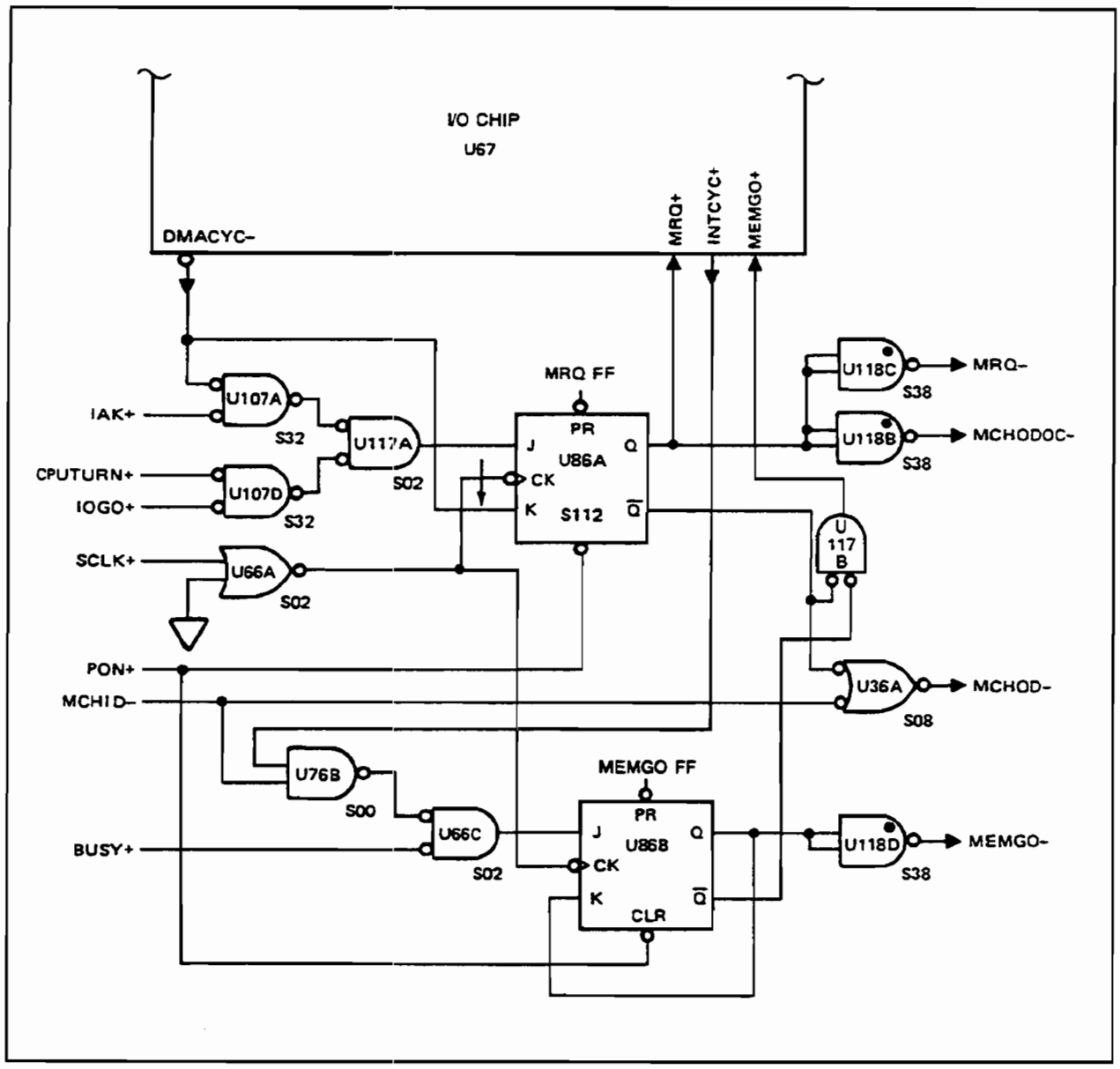
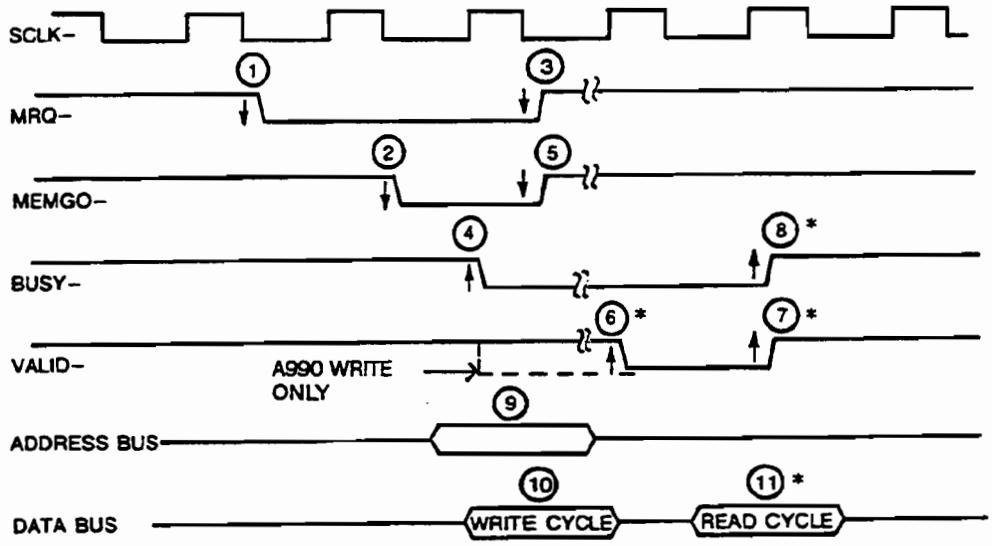


Figure 4-9. Memory Request Logic



10. An interface card asserts MRQ- to request a memory cycle. (MCHOD- will be asserted simultaneously to hold off all lower priority cards).
11. An interface card asserts MEMGO-, if one cycle after the assertion of MRQ-, it still has priority; i.e., its MCHID- is high. If MRQ- is not asserted, the MEMGO - is from the processor card, or an interface card during an interrupt cycle.
12. An interface card releases MRQ- at the end of the short half cycle when MEMGO- is released.
13. The memory asserts BUSY-, once MEMGO- has been asserted, in order to hold off other memory cycles until this cycle can be completed.
14. MEMGO- is released one cycle after being asserted.
15. The memory asserts VALID- during the last cycle of BUSY-.
16. The release of VALID- signals that data is valid on backplane.
17. The release of BUSY- signals that a new memory cycle can begin.
18. The address bus is driven by the interface card during the assertion of MEMGO-.
19. In the case of a memory write cycle, the interface data is valid on the backplane shortly after the address bus.
20. In the case of a memory read cycle, the memory guarantees valid data on the trailing edge of VALID.

*NOTE: A three-cycle memory access as used in an L-Series computer is shown above; A-Series computers use a two-cycle memory access, which causes events 6, 7, 8, and 11 to occur one state earlier.

Figure 4-10. Memory Handshake Timing

Interrupt Logic

Interrupt Signal Description

The I/O Master asserts INTRQ– to request an interrupt and ICHOD– to inhibit interrupt requests from lower priority devices. An interrupt request occurs when an IRQ signal from the user interface is asserted causing the IOP chip's FLAG FF to set while the I/O chip's CONTROL FF is set. IEN– and PRDIS– become active (Figure 4-14).

IEN– causes INTRQ– to be issued to the processor, and when the processor is ready to service the interrupt, it will acknowledge the request by asserting IAK+. INTRQ– is released when the card issues MEMGO).

PRDIS– pulls on ICHOD– to disable all lower-priority cards from requesting interrupt service. If a higher-priority card preempts the request, ICHID– will go low, disabling the requesting card. The lower-priority card should maintain its request until ICHID goes back up and the card can be serviced.

Interrupt Timing

Interrupt timing is shown in Figure 4-15.

Service Request and Mask Logic

A service request (SRQ–) from the user interface is used to initiate a data transfer. The service request mask signal (SRQM+) is used to inhibit SRQ– during diagnostic testing in diagnose mode 7. SACK– also inhibits SRQ– until the current service request is completed.

When power is applied to the system, PON– is deasserted, causing reset (RST– Figure 4-16) to be asserted, latching U106A and U106B in the reset condition with U106B-pin 6 low. Thus, SRQM+ is deasserted, and any assertion of service request (SRQ–) will drive U66-pin 4 high and asserts chip service request (CHSRQ–) at U66-pin 13 to the IOP chip. When diagnose mode 7 is entered, the signal DIAG– from the IOP to U106-pin 5 sets the latch and asserts SRQM+. This disables U66 so that any assertion of SRQ– will not be able to assert CHSRQ–. SRQM+ is also sent to the user interface as a utility signal defined by the I/O designer; for example, it may be used as a diagnostic loopback test enable signal. SRQM+ will be deasserted by a CLC 0 or on the next power-up.

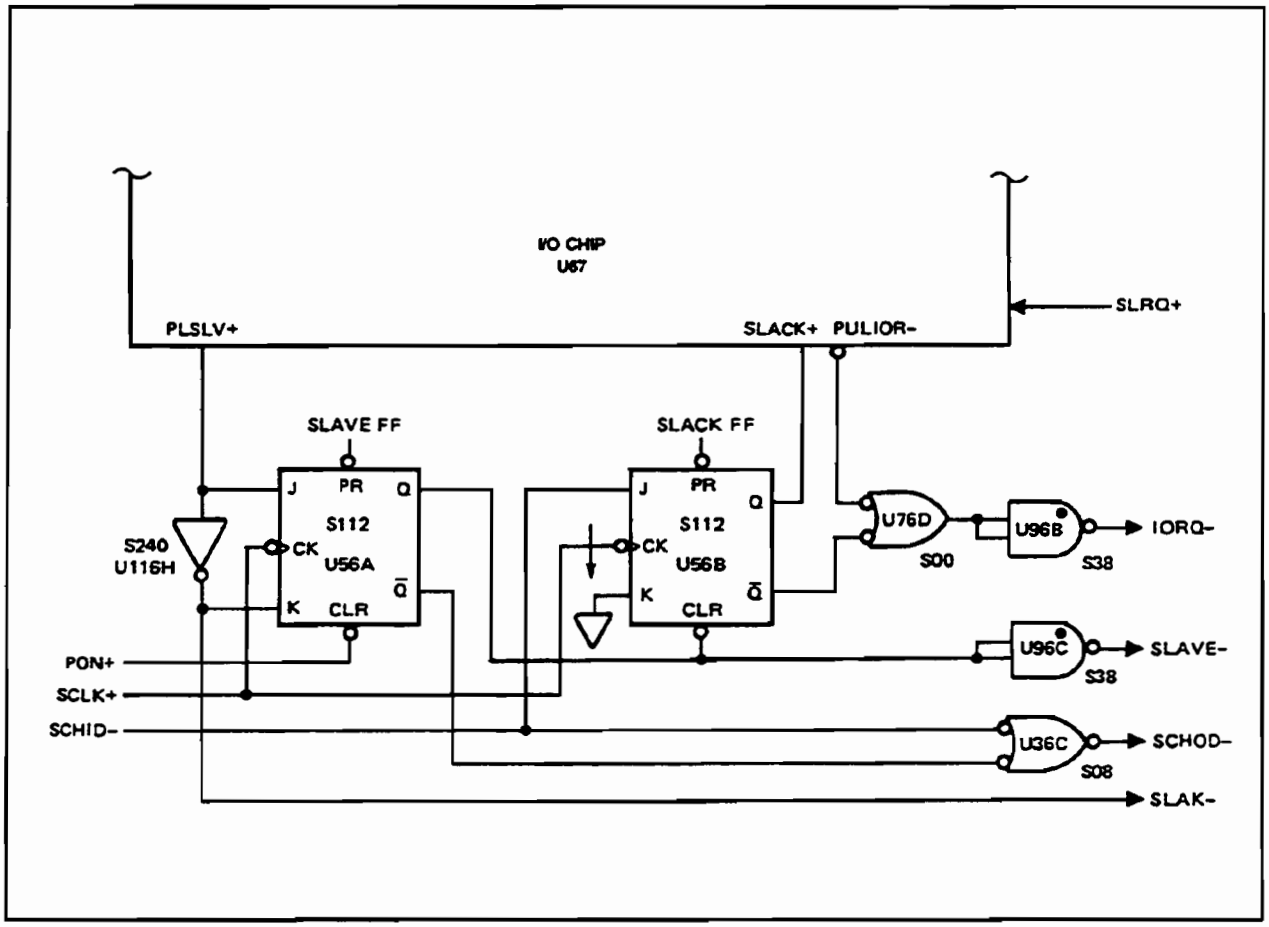
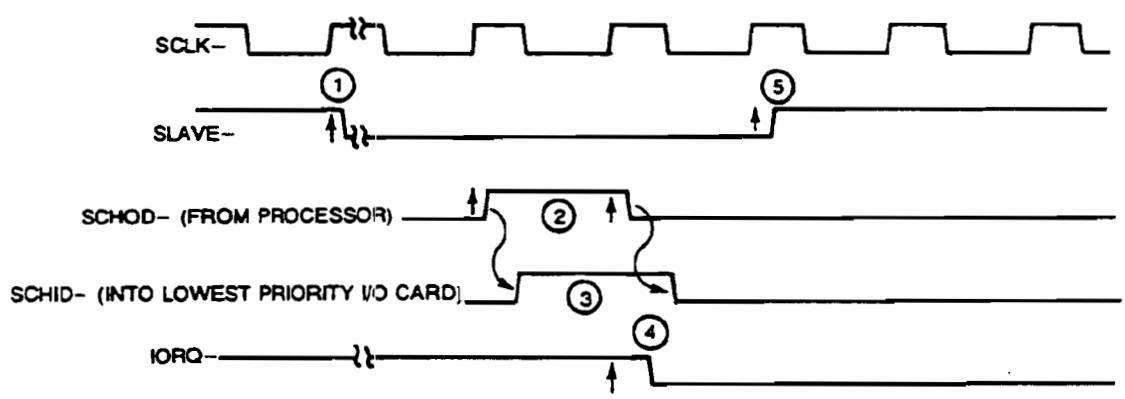
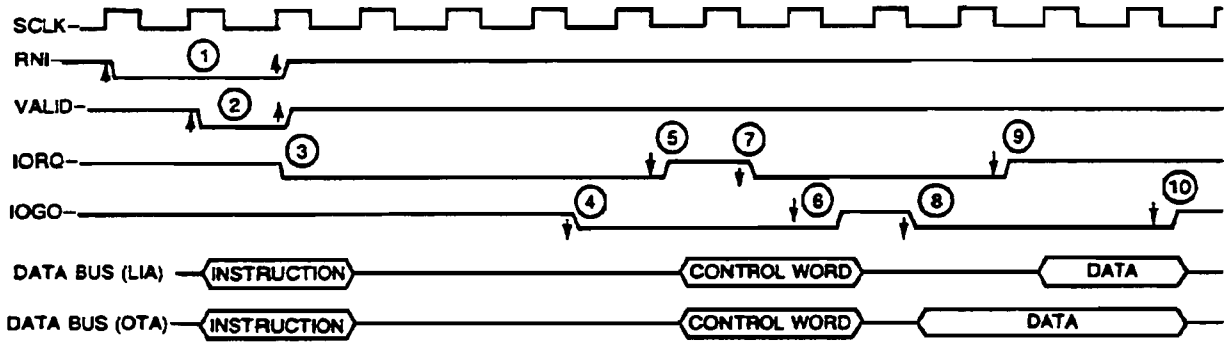


Figure 4-11. Slave and Handshake Logic



1. An interface card asserts SLAVE– to request the processor to enter slave mode.
2. When the processor has completed executing the current instruction, it acknowledges the assertion of SLAVE– by deasserting SCHOD– for one cycle.
3. Worse case, the SCHID/SCHOD priority chain has propogated down to the lowest-priority interface card by the end of that cycle, so that the last SCHID– will go high for one cycle.
4. The interface card received the enabling signal when its SCHID– signal went high, and can now pull on IORQ– in order to initiate the I/O handshake. The rest of the I/O handshake can then proceed exactly as shown in Figure 4-13.
5. The interface card deasserts SLAVE– once it has asserted IORQ–.

Figure 4-12. Slave Mode Timing



1. Processor asserts RNI- to inform all system cards that an I/O instruction is being fetched.
2. Memory asserts VALID- to inform all system cards that data on the backplane will soon be valid. Each interface should now latch the instruction off the data bus, and decode it to see if it is an I/O instruction to its select code.
3. An interface card pulls on IORQ- to signal that it recognized the I/O instruction and needs the CPU in order to execute it.
4. The processor asserts IOGO- to indicate that it is ready to receive a command from the interface card.
5. The interface card releases IORQ- to signal the processor that the control word will be available on the data bus on the second rising edge of SCLK-.
6. The processor releases IOGO- when it has clocked the command off the backplane.
7. The interface card reasserts IORQ- if another handshake is needed in order to transfer a data word.
8. The processor reasserts IOGO- in order to indicate that it is ready to receive an operand in the case of an input operation, or that data will be valid on next falling edge in the case of an output operation.
9. The interface card releases IORQ- to indicate that it has latched an operand off the backplane in the case of an output operation, or that an operand will be valid on the backplane on second rising edge in the case of an input operation .
10. The processor releases IOGO- to indicate that it has clocked data off the backplane in the case of an input operation, or that the handshake is complete in the case of an output operation.

Figure 4-13. I/O Handshake Timing

Reset Signals

There are several signals available from the I/O Master which are used to initialize the whole computer system or various parts of it. These are PON, RST, and in some cases BCS3. PON and RST reset the whole system. BCS3, asserted upon execution of an OTA 32, has been used in some applications as an I/O card reset. BCS3 is a minimum of 170 nanoseconds to a maximum of 254 nanoseconds, and as such can readily be used as a reset pulse.

I/O Master Diagrams

There are several diagrams useful for breadboard designing given in Figures 4-17 through 4-21. Included are a block diagram, schematic diagram, IC base diagrams, I/O breadboard assembly, I/O breadboard probe points, and a material list.

I/O Processor Chip Signal Definitions

Table 4-4 lists the pin assignments of the I/O processor chip. Table 4-5 provides a detailed definition of each signal, giving the function of each as well as the timing involved.

I/O Master-To-Device Interface

The A-Series I/O Master has a backplane interface and an I/O user interface. The signals available from the backplane to the I/O Master are described in the applicable Engineering and Reference Documentation Manual listed in Table 1-2. A detailed description of the signals available at the peripheral interface of the I/O Master for the interface designer is contained in Table 4-6. All signal lines to and from the user interface, except the DATA BUS, are buffered. The DATA BUS is not solely an I/O Master input, but is driven directly onto the backplane and is used for all system data transfers.

Backplane Clocks

There are two clocks available to the user interface. The Slow Clock (SCLK) is used by all system cards to synchronize all transfers that occur over the backplane. A fixed frequency Communications Clock (CCLK) is also available which may be used to drive state machines or may be divided down for baud rate generation.

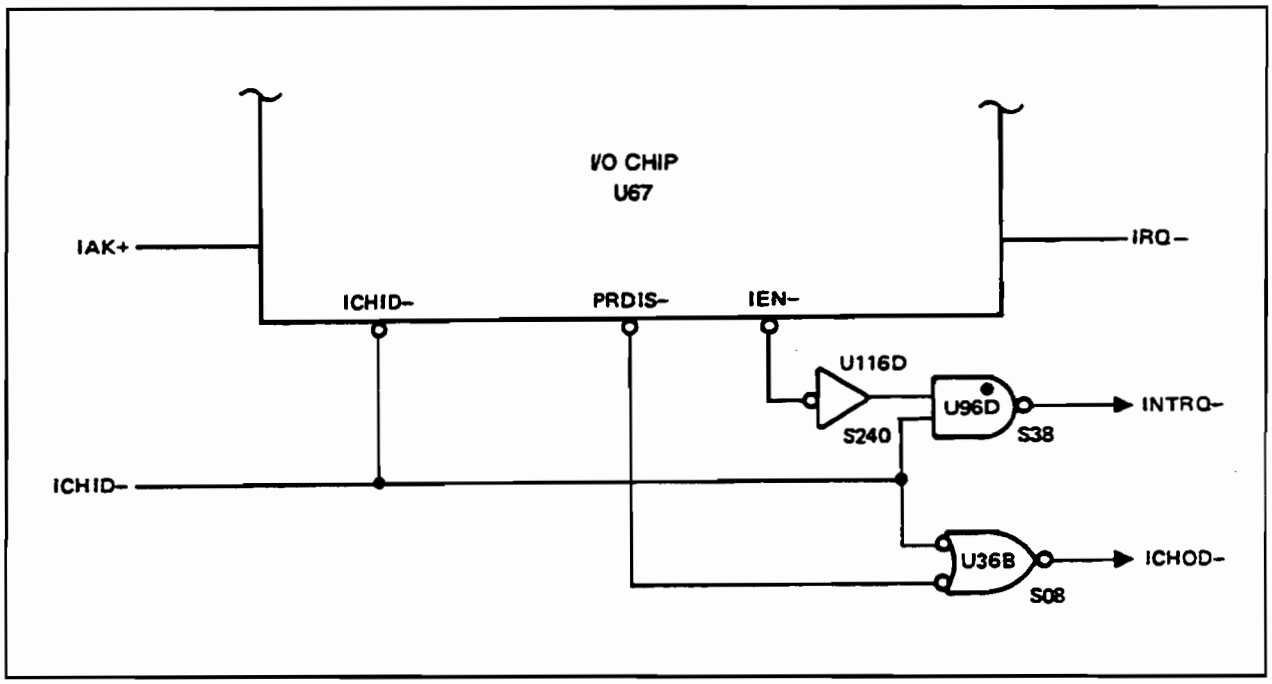
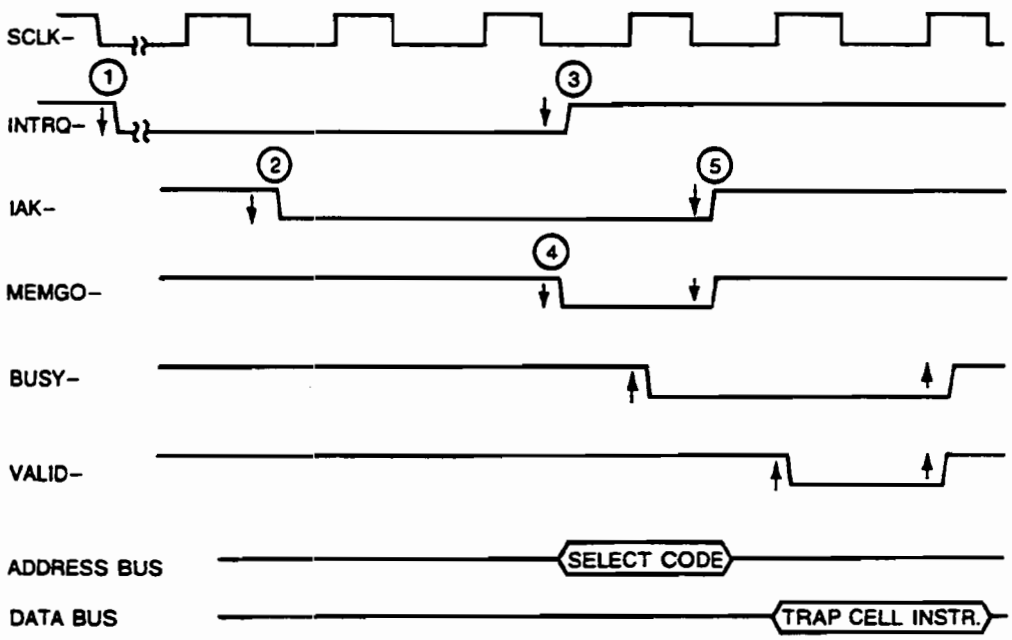


Figure 4-14. Interrupt Logic



1. An Interface card pulls on INTRQ- to request interrupt service.
2. When the processor has reached the appropriate state and if the interrupt system is enabled, and interrupts are not temporarily being held off, then it will acknowledge the interrupt request by asserting IAK-.
3. As soon as the interface card asserts MEMGO, it knows its interrupt will be serviced so it releases INTRQ-.
4. The interface card asserts MEMGO- to initiate a memory cycle, and during the one cycle of SCLK it holds MEMGO- low, it drives the lower 6 bits of the address bus with its select code, and the upper 9 bits with zeros.
5. The processor releases IAK- upon the assertion of BUSY-.

Figure 4-15. Interrupt Timing

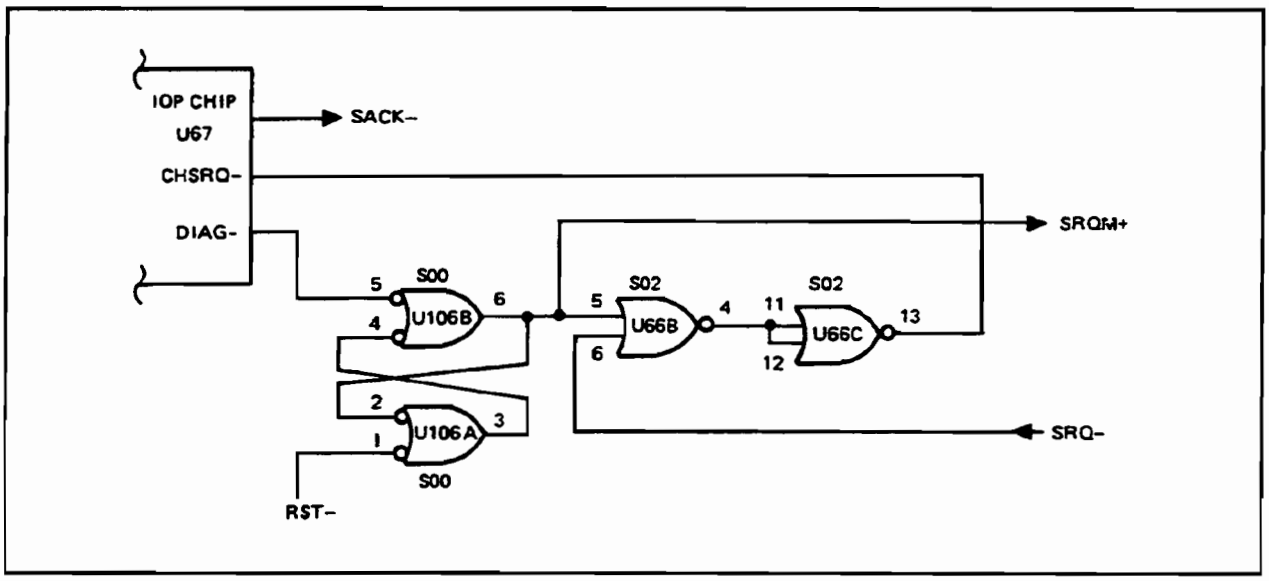


Figure 4-16. Service Request Mask Logic

Switch Options

The I/O Master includes an eight position rocker DIP switch (U1S1-S8). This switch pack is located near the front of the card, and the switches are used by both the I/O Master and the user interface. Six of the switches (S3–S8) are used for the select code. The select code is used only as a means of addressing the interface (and the peripheral to which it is connected), and bears no relation to the interrupt and DMA priority of the device. Switch S1 is used as a Virtual Control Panel select for those interfaces that use this feature. Switch S2 is user definable. It may, for example, be used to make the device command signal either active high (high true) or active low (low true).

Input Signals

There are four signals which may be input to the I/O Master by the user. These signals are IRQ, SRQ, SLRQ, and LSBYT; they are described in Table 4-6.

Output Signals

Excluding the data and power signals, there are 22 signals available on the I/O Master for use by the device interface or the device. These signals are described in Table 4-6. In addition to the signals described in Table 4-6, the following voltages are available at the user interface: +/–12V, +5V, and +5 VM. The memory voltage (VM) can be sustained by battery backup power in the event of power failure. The amount of time that memory can be retained is dependent on the current requirements, and must be calculated.

Signal Timing Specifications and Waveforms

The following paragraphs provide the timing specifications for all the signals that interact between the I/O Master and the user interface. The timing specifications of the signals between the I/O Master and the backplane are described in the applicable Engineering and Reference Documentation Manual listed in Table 1-2.

Table 4-7 contains the definitions of terms used in timing diagrams and timing specifications. All times are given in nanoseconds unless otherwise indicated.

Table 4-8 contains the I/O Master input timing requirements.

Output Timing Specifications

All signals provided to the user interface are listed in Tables 4-9 through 4-24 in alphabetical order. These specifications take into account delays due to capacitive loading. Of particular concern are the signals CFF, DMAEN, DVCMD, LOBYT, and SACK. These five signals are specified into a load of 30 picofarads. For larger capacitive loads, a delay of 0.4 ns/pf must be added.

Timing Diagrams

Figures 4-22 through 4-30 contain timing diagrams that show the interaction between the device specific I/O card logic and the I/O Master. DMA corner cases are shown in detail. In all these diagrams, any unlabled arrow is a reference edge to SCLK–. The timing specifications for SCLK– may be found in Table 4-23.

Designing an Interface to the I/O Master

An I/O interface PCA must provide the circuits through which data can be transferred between the computer and an external I/O device. It must also provide the circuits required to control the I/O device from commands received from the computer. A typical interface PCA may contain as many as 16 buffers for temporary storage of data both to and from the I/O device. The number of buffers contained on a particular interface PCA depends on its associated I/O device. Some I/O devices require the capability of interrupting the computer program while for others, this capability is not necessary. Some I/O devices require control signals for the movement of tape, etc., and some require special timing signals. There are many special cases in which unique types of controls or other criteria dictate the need to design and fabricate a special I/O interface PCA. Due to the very nature of special purpose interfacing, no detailed step-by-step procedures for the best design can be given. Only a study of the computer and I/O device mutual requirements can produce the ultimate design. Therefore, the information presented in this chapter should be used as guidelines around which to base your own interface design.

Designing an interface to the I/O Master can be as simple as hooking up two wires. Consider an application such as interfacing to a sensor or measurement device. This "input only" I/O card

would need a data register (commonly referred to as Register 30) and the ability to assert SRQ- (Service Request) to the I/O Master when it had data ready. When it is time to drive the data in Register 30 onto the backplane, the I/O Master asserts BCS5- (Bus Control State 5). These two signals SRQ- and BCS5- are sufficient to interface the sensor to the A-Series computer. An I/O card like this would only use a small subset of the I/O Master capabilities. Let us add capabilities to this card, and discuss the signals that will be required to implement these capabilities.

First, let us add a control register in order to allow the programmer to select various modes of operation. This Control register can be loaded in one of two ways. It may be loaded by executing an OTA 31 or during a DMA self-configuration. In either case, the signals BCS2- and CKDAT- (Clock Data) will be asserted by the I/O Master. These two signals may be used (NORed) to generate the load pulse for the control register.

Now let us add some handshaking capability to the interface. We want to allow the programmer to execute an instruction which will act as a start pulse for several sensors and cause them to take a reading. This signal is called DVCMD- (Device Command). It is asserted for one state (227 ns) by the I/O Master under two different circumstances. First, it will be asserted whenever an STC 30 is executed; secondly, during DMA it will be asserted (if programmatically selected in the DMA control word) for one state following each word/byte that gets written to the interface.

Let us assume that our sensors give us eight bits of data. If we were to DMA into memory in this manner we would waste half of each 16-bit word as well as taking twice as many backplane memory cycles as our I/O card really needs. Instead of this, we can use the I/O Master byte packing capability. The I/O Master assumes the convention that the upper byte is always transferred first. The signal LOBYT- (low byte), generated by the I/O Master, is used to route the data to the proper section of the data register. When LOBYT- is high, data should be written into the upper byte of the data register (Register 30). When LOBYT- is low, data should be written into the lower byte of the data Register (register 30).

By combining all of the above control and handshake signals, we have created a simple interface that will acquire and transfer data efficiently. The actual transfer to memory may be by programmed I/O or high speed DMA, depending on how the driver is written for this particular application.

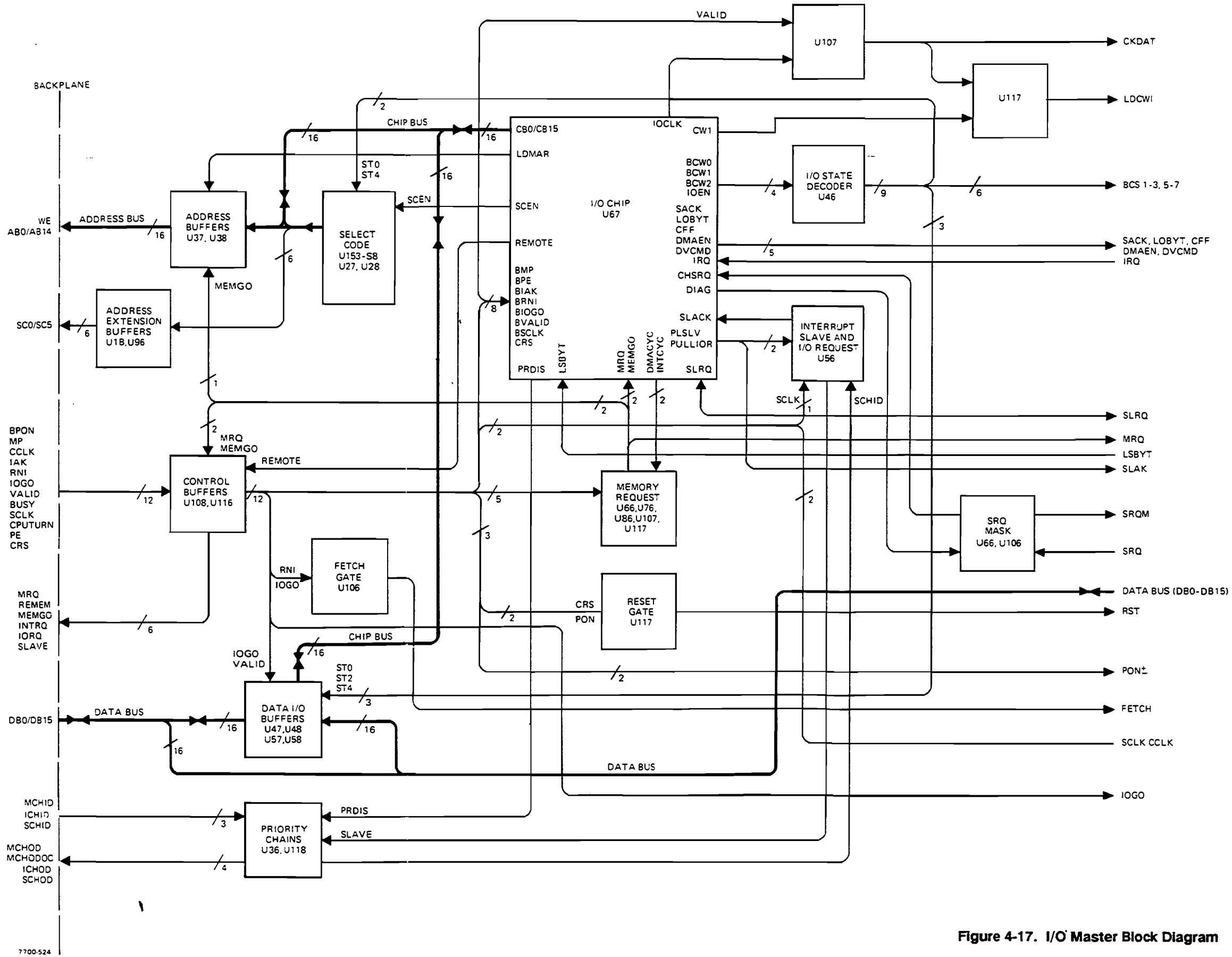


Figure 4-17. I/O Master Block Diagram

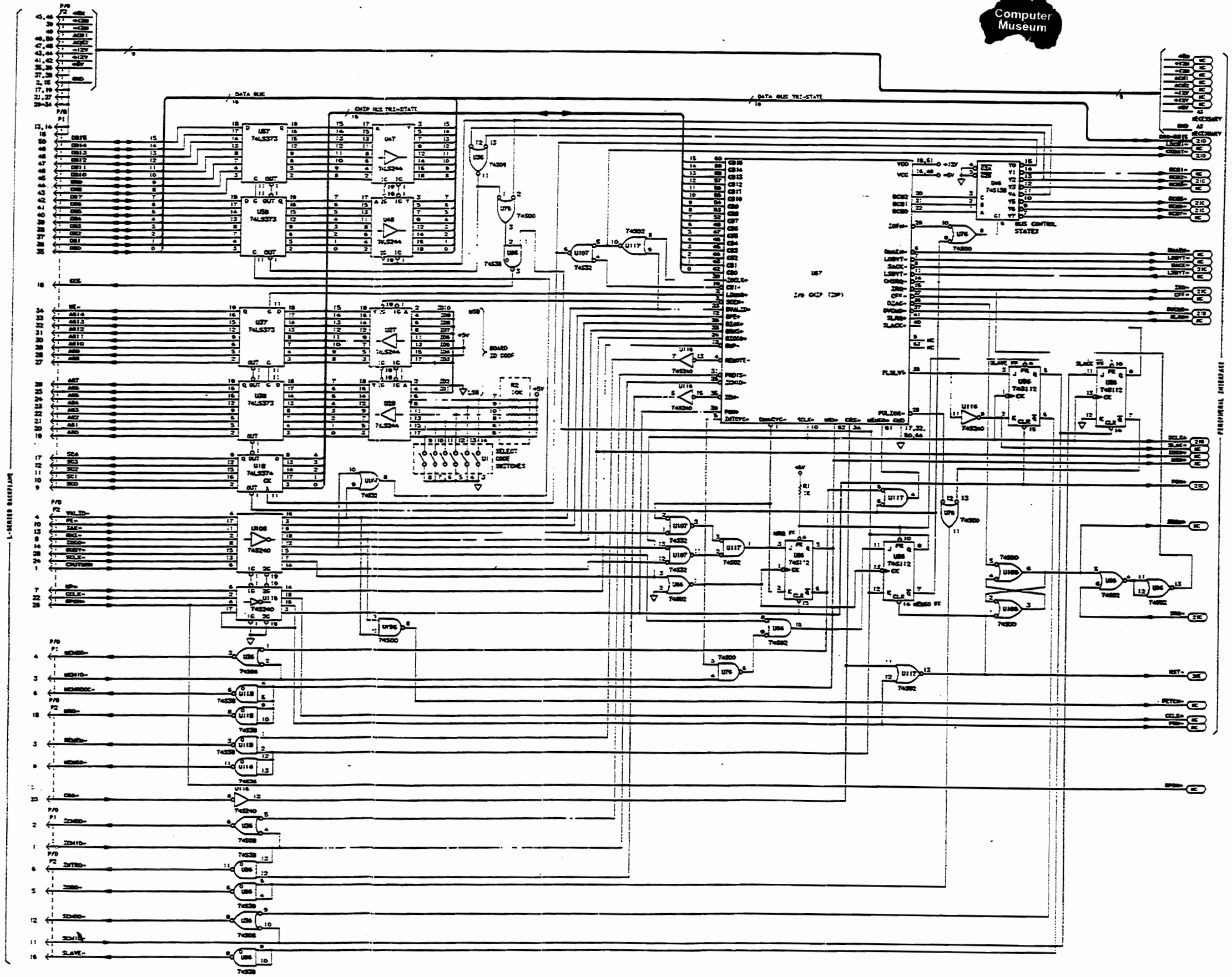


Figure 4-18. HP12010-60003 I/O Master Schematic Diagram

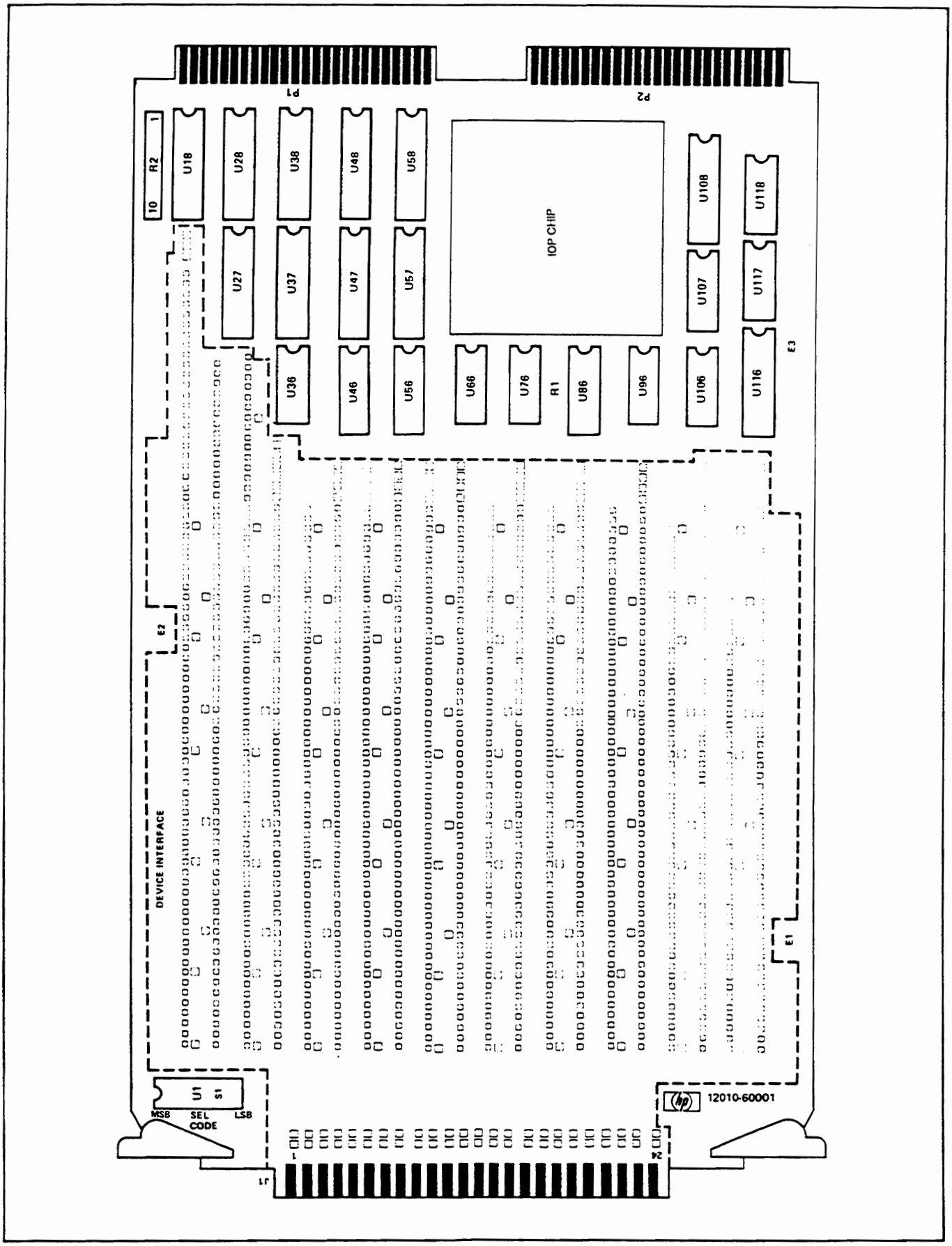


Figure 4-19. HP 12010-60003 I/O Breadboard Assembly

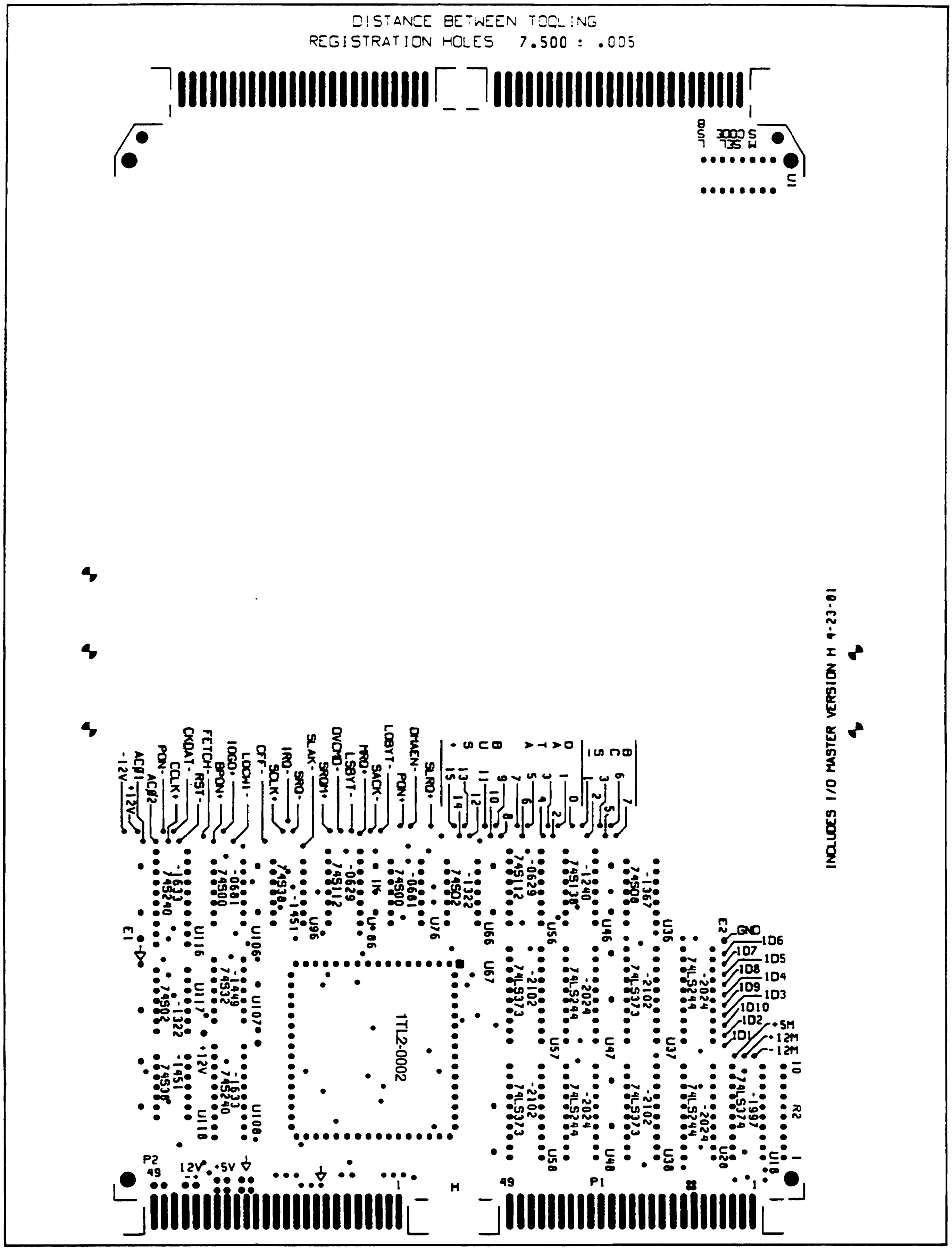
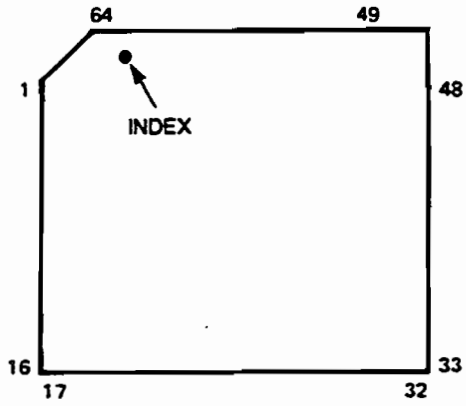


Figure 4-20. HP 12010-60003 I/O Breadboard Probe Points

Table 4-3. HP 12010-60003 I/O Master Material List

Reference Designation	HP Part Number	Description	Purchasing Information
	12010-60003	ASSEMBLY, I/O BREADBOARD	
E1-E3	0360-1682	STANDOFF, single turret single-ended standoff	Litton Precision Products Inc. USECO Div., part no. 14758
	0403-0289	EXTRACTOR-PC BOARD .063" Thickness, 1.122" Long, 0.315" Wide	HI-PRO Mfg. Inc.
R1	0757-0280	RESISTOR, 1k, 1%, 0.125W	TRW Inc. Burlington Div., CEA-993
XU67	1200-0981	SOCKET, 64 Pin	
R2	1810-0280	NETWORK-RES, 10-Pin-SIP 9x10k, 2%, 0.125W	Allen Bradley Co., part no. 210A103
U1S1-S8	3101-2243	SWITCH, DIP 8-Rocker 0.05A @ 30Vdc	Grayhill Inc., part no. 76PSB08
U18	1820-1997	IC FF, D-Type Octal TTL Tri-state	SN74LS374PC
U27-28	1820-2024	IC BUFFER/LINE DRIVER, Octal, TTL Tri-State	SN74LS244N
U36	1820-1367	IC GATE, Quad 2-Input Positive AND TTL	SN74S08N
U37-38	1820-2102	IC LATCH, D-Type Octal TTL Tri-state	SN74S373N
U46	1820-1240	IC DECODER, 3-to-8 Line TTL	SN74S138N
U47-48	1820-2024	IC BUFFERULINE DRIVER, Octal, TTL Tri-state	SN74S244N
U56	1820-0629	IC FF, Dual J-K Neg-Edge-Triggered TTL	SN74S112N
U57-58	1820-2102	IC LATCH, D-Type Octal TTL Tri-state	SN74S373N
U66	1820-1322	IC GATE, Quad 2-Input Positive NOR TTL	SN74S02N
U67	1TL2-0002	IC I/O Processor Chip	1TL2-0002
U76	1820-0681	IC GATE, Quad 2-Input Positive NAND TTL	SN74SOON
U86	1820-0629	IC FF, Dual J-K Neg-Edge-Triggered TTL	SN74S112N
U96	1820-1451	IC GATE, Quad 2-Input Positive NAND Buffer OC	SN74S38N
U106	1820-0681	IC GATE, Quad 2-Input Positive NAND TTL	SN74SOON
U107	1820-1449	IC GATE, Quad 2-Input Positive OR TTL	SN74S32N
U108	1820-1633	IC BUFFER/LINE DRIVER, Octal, TTL Tri-state	SN74S240N
U116	1820-1633	IC BUFFER/LINE DRIVER, Octal, TTL Tri-state	SN74S240N
U117	1820-1322	IC GATE, Quad 2-Input Positive NOR TTL	SN74S02N
U118	1820-1451	IC GATE, Quad 2-Input Positive NAND Buffer OC	SN74S38N

Table 4-4. I/O Processor Chip Pin Definitions



I/O PROCESSOR (IOP) CHIP OUTLINE (COMPONENT SIDE)

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
1	DMACYC-	17	GND	33	BRNI+	49	VCC
2	LDMAR+	18	VDD	34	CRS+	50	GND
3	SCEN-	19	CW1-	35	IEN-	51	VDD
4	REMOTE-	20	BCW2+	36	DIAG-	52	CB7+
5	INTCYC+	21	BCW1+	37	DVCMD-	53	CB8+
6	DMAEN-	22	BCW0+	38	PLSLV+	54	CB9+
7	LOBYT-	23	BVALID+	39	PON+	55	CB10+
8	SACK-	24	BIOGO+	40	SLACK+	56	CB11+
9	NC	25	ICHID-	41	SLRQ+	57	CB12+
10	SCLK+	26	BIAK+	42	CB0+	58	CB13+
11	LSBYT-	27	CFF-	43	CB1+	59	CB14+
12	BPE+	28	PULIOR-	44	CB2+	60	CB15+
13	BMP-	29	IOEN-	45	CB3+	61	MEMGO+
14	CHSRQ-	30	IOCLK+	46	CB4+	62	MRQ+
15	IRQ-	31	PRDIS-	47	CB5+	63	NC
16	VCC	32	GND	48	CB6+	64	GND

I/O Interface Design

The information presented thus far in this manual has been to help you become thoroughly familiar with the A-Series I/O system in general, and with the I/O Master in particular. Chapter 3 included a synopsis of the capabilities of some of the A-Series I/O cards. If it appears that one of them could serve your purpose, then you should obtain the reference manual (listed in Table 1-2) for that interface for a more detailed description. If the presently available interface cards do not meet your needs then you should start your design using the standard I/O Master, because circuit timing constraints are minimized, and a great deal of effort has been expended to minimize layout, parts count, etc. Consider that changes in the I/O Master area affect self-test, diagnostics, and other built-in computer features. The specifications provided (loading, timing, etc.) are for the standard I/O Master and breadboard card. Using any other design may cause these specifications to vary.

Table 4-5. I/O Processor Chip Signal Definitions

PIN NO.	SYMBOL	
1	DMACYC–	**Name:** DMA Cycle **Function:** This output is used in conjunction with external logic to provide the MRQ+ (Memory Request) signal to both the I/O Chip and the memory priority chain in order to do a DMA transfer. **Timing:** During DMA transfers, the signal is asserted on the second positive transition of SCLK+ (Slow Clock) following the assertion of SRQ– (Service Request) with the contingency that the last assertion of VALID+ must have occurred one or more positive transitions of SCLK+ prior to SRQ–. The signal is released on the last transition of SCLK+ following the assertion of MEMGO+ (Memory Go). During a self-configuration, the signal is asserted by the DMA state machine. The signal is released by MEMGO+ during the last memory cycle of the configuration.
2	LDMAR+	**Name:** Load Memory Address Register **Function:** This output is used to gate the Chip Bus outputs, CB0+ through CB15+, into latches/flops external to the I/O Chip (CB0+ is LSB). **Timing:** During DMA transfers the signal is asserted by MRQ+ and released by the DMA state machine. When a self-configuration DMA transfer is initiated, the first assertion and release of the signal will be as before. All subsequent assertions and releases will be controlled by the DMA state machine. During an interrupt sequence, the signal is asserted on the first negative transition of SCLK+ after the assertion of IAK+ (Interrupt Acknowledge) and released on the second subsequent positive transition of SCLK+.
3	SCEN–	**Name:** Select Code Enable **Function:** This output provides a gating signal to allow external logic to place the Select Code of the chip and a card identification number on the Chip Bus bits CB0+ through CB15+ during power up (PON) or restart sequences. **Timing:** The signal is asserted and de-asserted by the respective de-assertion and assertion of PON+ (Power On). Additionally, the signal is asserted and de-asserted by CRS+ (Control Reset). With the assertion of PON+ only, and not CRS+, the chip latches the Chip Bus data into the SC and ID Register.
4	REMOTE–	**Name:** Remote Memory **Function:** This output is used to enable or disable the remote memory, if used. **Timing:** The signal is asserted and de-asserted by the DMA state machine when DMA is enabled provided IAK+ has not been asserted and Bit 10 (REMEM+) of CW1 (Control Word 1 Register) has been set (i.e., during an OT*21).
5	INTCYC+	**Name:** Initiate Cycle **Function:** This output, in conjunction with external logic, is used to generate MEMGO+ (Memory Go) which starts a memory cycle. **Timing:** During DMA transfers the signal is asserted by the first positive transition of SCLK+ following the assertion of MRQ+ and released by the DMA state machine, with the assertion of MEMGO+. When the DMA self-configuration is initiated, the first assertion and release of the signal is as before. All subsequent assertions are controlled by the DMA state machine. During an interrupt sequence, the signal is asserted on the second positive transition of SCLK+ following the assertion of IAK+. The signal is released on the next positive assertion of SCLK+.
6	DMAEN–	**Name:** DMA Enable **Function:** This output indicates DMA is enabled. **Timing:** The signal is asserted after a DMA self-configuration, or by the execution of an OT*23 and an STC21. The release of the signal occurs when the last SACK– (Service Request Acknowledge) is asserted on DMA input transfers or when the next to the last SACK– is asserted on DMA output transfers. The signal is also de-asserted by the de-assertion of PON+ or the assertion of CRS+, CLC22, CLC 23, or PE+ (Parity Error) if during a memory cycle after VALID+ has been asserted.

Table 4-5. I/O Processor Chip Signal Definitions (Continued)

PIN NO.	SYMBOL	
7	LOBYT–	**Name:** Low Byte **Function:** This output is used to implement byte mode DMA transfers where bytes are packed into 16 bit words. The assertion of the signal is used to enable data into the lower byte of the word. The de-assertion of the signal is used to enable data into the upper byte. Additionally, the signal is asserted if DMA is not enabled or if word mode DMA transfers are executed. **Timing:** The signal asserts or de-asserts after each assertion of SACK–, provided the assertion of SACK– is due to a DMA transfer.
8	SACK–	**Name:** Service Request Acknowledge **Function:** This signal is used to acknowledge a service request and allow the initiation of additional service requests. **Timing:** The signal is asserted by the first positive transition of SCLK+ occurring after the first negative transition of SCLK+ during which, SRQ– was asserted. The signal is de-asserted by the first positive transition of SCLK+ following the first negative transition of SCLK+ after the de-assertion of SRQ–. The signal may also be asserted/de-asserted by SCLK+ at the beginning of a DMA output transfer if the AUTO+ bit (Bit 8) of CW1 was set during configuration.
9		NOT USED
10	SCLK+	**Name:** Slow Clock **Function:** This input is the primary timing/synchronizing signal used by the chip. **Timing:** Period about 227 ns for A400/A600 (250 ns for A700); duty cycle about 60%; about 267 ns for A900 (80% Duty Cycle); 250 ns for A990.
11	LSBYT–	**Name:** Last Byte **Function:** This input is used to terminate a DMA transfer before word count rollover occurs. The assertion of the signal will also set Flag 21 (test with a SFS21 or SFS23) unless the DMA transfer was a self-configuration with Bit 15 (CONT) of CW1 set. In this case, the chip will automatically configure the next transfer. The assertion of LSBYT– (or word count rollover) during the last self-configured DMA transfer will set both Flag 20 and Flag 21. Subsequently, if interrupts from the chip are not masked and if Bit 11 (CINT+) of CW1 is not set then IEN– (Interrupt Enable) and PRDIS– (Priority Disable) will be asserted. **Timing:** The signal must be asserted on or before the first negative transition of SCLK+ following the assertion of SRQ– which requests the last DMA transfers.
12	PE+	**Name:** Parity Error **Function:** This input is used to indicate the occurrence of a parity error during a memory read. **Timing:** The assertion of the signal during an instruction fetch cycle while RNI+ (Read Next Instruction) or VALID+ is asserted and including the SCLK+ assertion interval immediately following the release of VALID+ will prevent the assertion of IORQ– (Input/Output Request), BCW0+, BCW1+ (Bus Control Words 0 and 1 respectively), and CW1– (Control Word 1), as well as the alteration of the internal Flags (20, 21, 22, and 30) and Controls (3, 20, 21, 22, 23, and 30). The assertion of the signal during a DMA transfer, while VALID+ is asserted and including the SCLK+ assertion interval immediately following the release of VALID+, will de-assert DMAEN– and DMACYC+ and reset the DMA state machine to its initialization state, state 1.
13	MP–	**Name:** Memory Protect **Function:** This input is used to prevent the assertion of IORQ–, BCW0+, BCW1+, CW1–, and the alteration of the internal Flags (20, 21, 22, and 30) and Controls (3, 20, 21, 22, 23, and 30) unless IAK+ is asserted. **Timing:** This signal is asynchronous with respect to all other chip pins, but should change states only during the assertion of RNI+.

Table 4-5. I/O Processor Chip Signal Definitions (Continued)

PIN NO.	SYMBOL	
14	SRQ–	**Name:** Service Request **Function:** This signal is used to initiate data transfers. Under program control, the assertion of SRQ+ will set Flag 30 if the chip is not in an instruction fetch handshake sequence. Once Flag 30 is set, SACK+ (Service Request Acknowledge) is asserted. During DMA transfers, the assertion of the signal is used to request a word/byte transfer. **Timing:** This signal is clocked into the chip on the negative transition of SCLK+.
15	IRQ–	**Name:** Interrupt Request **Function:** This input is used to set Flag 30 if the chip is not in an instruction fetch handshake sequence, thereby indicating an interrupt condition through the assertion of IEN– and PRDIS–, provided that interrupts are not masked and Control 30 is set. Note: If Control 30 is set, CFF– (Control Flip-Flop) will be asserted. **Timing:** The signal is clocked into the chip on the negative transition of SCLK+.
16, 49	VCC	**Name:** VCC **Function:** +5 Volt Power Supply **Timing:** None
17, 32, 50, 64	GND	**Name:** Ground **Function:** 0 Volt Power Supply **Timing:** None
18, 51	VDD	**Name:** VDD **Function:** +12 Volt Power Supply **Timing:** None
19	CW1	**Name:** Load DMA Control Word 1 **Function:** This output is used to indicate the presence of DMA Control Word 1 on the Chip Bus. **Timing:** This output is asserted on the first negative transition of SCLK+ following the first assertion of IOGO+ (I/O GO) occurring after the assertion of IORQ–, in response to an OUT 21 instruction. The signal de-asserts 2 SCLK+ periods after the assertion or on the first negative transition of SCLK+ following the de-assertion of IOGO+, whichever occurs first. During DMA self-configuration the signal is asserted by the first positive transition of SCLK+ following the first assertion of LDMAR+ and de-asserts on the first positive assertion of SCLK+ after the de-assertion of the first VALID+.
20	BCW2+	**Name:** Bus Control Word Bit 2 **Function:** This output is the MSB of the 3 bit bus control word used to control data transfers between chip and external logic. **Timing:** The signal is asserted on the negative transition of SCLK+ following the assertion of IOGO+ (I/O GO) occurring after the assertion of IORQ–. The signal de-asserts on the second SCLK+ period after the assertion or on first negative transition of SCLK+ following the de-assertion of IOGO+, whichever occurs first. During DMA transfers, the signal is asserted by the first positive transition of SCLK+ following the assertion of MRQ+ or by MRQ+, if MRQ+ is asserted while SCLK+ is asserted (MRQ+ is enabled into the chip during the assertion of SCLK+), whichever occurs last. For DMA input transfers, the signal is de-asserted on the first positive transition of SCLK+ following the de-assertion of MRQ+. However, for DMA output transfers, the signal is de-asserted on the first positive assertion of SCLK+ following the de-assertion of VALID+.

Table 4-5. I/O Processor Chip Signal Definitions (Continued)

PIN NO.	SYMBOL	
21	BCW1+	**Name:** Bus Control Word Bit 1 **Function:** This output is the LSB+ 1 bit of the Bus Control Word used to control data transfers between the chip and external logic. **Timing:** Same as BCW2+.
22	BCW0+	**Name:** Bus Control Word Bit 0 **Function:** This output is the LSB of the Bus Control Word. **Timing:** Same as BCW2+.
23	VALID+	**Name:** Valid **Function:** This input is used to latch instructions in the chip and provide a control input to the DMA state machine. **Timing:** This signal is clocked into the chip on the positive transition of SCLK+.
24	IOGO+	**Name:** I/O Handshake Request Acknowledge **Function:** This input is used to handshake with a processor via the IORQ− output and provide control of the Bus Control Word, IOEN− (I/O Enable) and IOCLK+ (I/O Clock) outputs. This signal is normally asserted in response to the assertion of IORQ− and will subsequently cause the de-assertion of IORQ−. Therefore, IOGO+ must be de-asserted in order to re-assert IORQ−, such as in multiple I/O handshake operations. **Timing:** The signal is clocked into the chip on the first negative assertion/de-assertion of IOGO+.
25	ICHID−	**Name:** Interrupt Chain In Disable **Function:** This input is asserted to inhibit the assertion of INTCYC+ during the assertion of IAK+. Also, in Diagnose Modes 1 and 2, the signal inhibits the assertion of IORQ−. **Timing:** The signal may be asserted or deasserted anytime, except during the assertion of IAK+. In Diagnose Modes 1 and 2, the signal must be asserted or de-asserted on or before the negative transition of SCLK+ which follows the second positive transition of SCLK+ occurring after VALID+ is de-asserted during an instruction fetch.
26	IAK+	**Name:** Interrupt Acknowledge **Function:** This input is used to read the Select Code of the chip, and initiate the trap cell fetch. **Timing:** This signal is clocked into the chip on the negative transition of SCLK+. On the first negative transition after the assertion of the signal, the chip will output its Select Code on CB0+ through CB5+ (Chip Bus Bits 0 through 5) and 0's on CB6+ through CB15+. LDMAR+ will be asserted. On the next positive transition of SCLK+, INTCYC+ will be asserted. IEN− will be de-asserted on the first negative transition of SCLK+ after the de-assertion of IAK+ or the third negative transition of SCLK+ after the assertion of IAK+, whichever occurs first.
27	CFF−	**Name:** Control Flip-Flop **Function:** This output is used to indicate the state of the control flip-flop and is set to enable IEN− via Flag 30. **Timing:** This signal is asserted with the execution of an STC30 on the first negative transition of SCLK+ following the second positive transition of SCLK+ after the de-assertion of VALID+. The signal is de-asserted with the execution of a CLC30 or a CLC0 the assertion of CRS+, or the de-assertion of PON+.

Table 4-5. I/O Processor Chip Signal Definitions (Continued)

PIN NO.	SYMBOL	
28	IORQ−	**Name:** I/O Request **Function:** This output is asserted to initiate an I/O handshake cycle with the CPU. It is de-asserted to acknowledge the assertion of IOGO+ from the CPU. **Timing:** During an I/O instruction execution, the initial assertion of this signal is via an instruction decode. Likewise, during a SLAVE/BREAK mode cycle, the signal is asserted in response to the assertion of SLACK+. All subsequent assertions and de-assertions are in response to IOGO+ and are caused by the positive going transitions of SCLK+. After the initial assertion of the signal, it is de-asserted by the first positive transition of SCLK+ following the first negative transition of SCLK+ after the assertion of IOGO+. If an additional handshake is required, the signal is re-asserted on the next (second) positive transition of SCLK+. The signal remains asserted until IOGO+ is de-asserted and subsequently re-asserted.
29	IOEN−	**Name:** I/O Enable **Function:** This output is used to enable the Bus Control Word decoding external to the chip. **Timing:** During an instruction fetch, the signal is asserted by the assertion of RNI+ and de-asserted by the first negative transition of SCLK+ following the first positive transition of SCLK+ occurring after the de-assertion of VALID+. For I/O handshake sequences, the signal is asserted during the IOGO+ handshake cycle by the first positive transition of SCLK+ following the first negative transition of SCLK+ occurring after the assertion of IOGO+. The signal is de-asserted by the second subsequent negative transition of SCLK+. During DMA output transfers the signal is asserted by the first positive transition of SCLK+ following the assertion of MEMGO+ and de-asserted by the first positive transition of SCLK+ following the first negative transition of SCLK+ occurring after the de-assertion of VALID+.
30	IOCLK+	**Name:** I/O Clock **Function:** This output is used to provide a clock or latch signal to capture the data bus using external flip-flop or latches. **Timing:** This signal is asserted during the second handshake cycle of an OT* instruction sequence by the first positive transition of SCLK+ following the first negative transition of SCLK+ occurring after the assertion of IOGO+. The signal is de-asserted on the next positive transition of SCLK+.
31	PRDIS−	**Name:** Priority Disable **Function:** This output is used to disable lower priority devices from causing interrupts. **Timing:** During normal operation, this signal is asserted by the first positive transition of SCLK+ occurring after the setting of the appropriate Flags and Controls (i.e., Flag 30 and Control 30) and if interrupts are not masked as determined by the IMR (Interrupt Mask Register) and Chip Select Code (Select Code and ID Register — Bits 0 through 5). The signal is de-asserted by the first positive transition of SCLK+ following the clearing of the appropriate Flag and/or Control, the assertion of CRS+, or the de-assertion of PON+. When in Diagnose Modes 1 or 2, the signal is asserted during the second assertion of IOGO+ while executing the OT* 02 which initiates the Diagnose Mode. The signal is de-asserted during the second assertion of IOGO+ while executing an I/O handshake sequence in response to an LI* 02 instruction.
33	RNI+	**Name:** Read Next Instruction **Function:** This input is asserted to indicate an instruction fetch and causes the chip to assert IOEN− and enables the internal IRL (Instruction Register Latch) to receive the fetched instruction. **Timing:** The signal is asserted during an instruction fetch and clocked into the chip on the first positive transition of SCLK+ after the assertion of the signal.
34	CRS+	**Name:** Control Reset **Function:** This input is used to reset the Flag and Control functions and the DMA state machine. **Timing:** The signal may be asserted asynchronously but must be at least one period of SCLK+ duration.

Table 4-5. I/O Processor Chip Signal Definitions (Continued)

PIN NO.	SYMBOL	
35	IEN–	**Name:** Interrupt Enable **Function:** This output is used to indicate an interrupt request. **Timing:** The signal is asserted by the first positive transition of SCLK+ occurring after the setting of the appropriate Flags and Controls (i.e., Flag 30 and Control 30) and if interrupts are not masked.
36	DIAG–	**Name:** Diagnose **Function:** This output is used to enable external logic to perform diagnostic testing. **Timing:** The signal is asserted by the third negative transition of SCLK+ following the second assertion of IOGO+ occurring after the execution of an OT*2 where A=7. The signal is de-asserted by the execution of an OT*2 where A<7, the assertion of CRS+, or the de-assertion of PON+.
37	DVCMD–	**Name:** Device Command **Function:** This output is used to initiate a data transfer. **Timing:** The signal is asserted by the third positive transition of SCLK+ in response to the execution of an STC 30. The signal is de-asserted on the next positive transition of SCLK+. During DMA, the signal is asserted for each data transfer if the STC bit (Bit 14) of DMA Control Word 1, CW1 is set. Again the signal is asserted/de-asserted on the positive transition of SCLK+ and lasts for one period. For DMA input transfers, the signal is asserted following the assertion of MEMGO+ and for DMA output transfers it is asserted following the de-assertion of VALID+.
38	PLSLV+	**Name:** Pullslave **Function:** This output is used to acknowledge the receipt of SLRQ+ (Slave Request) and to request I/O handshakes. **Timing:** The signal is asserted by the assertion of SLRQ+ if at PON breaks were enabled and if an STC2 has been executed.
39	PON+	**Name:** Power On **Function:** This input is used to provide a chip reset and to indicate power is on. **Timing:** When not asserted, the signal will reset the chip. On the positive transition, the data present on the Chip Bus bits CB0+ through CB15+ are clocked into the SC and ID Register.
40	SLACK+	**Name:** Slave Acknowledge **Function:** This input is used to acknowledge the occurrence of a slave mode request as indicated by the assertion of PLSLV+. **Timing:** The signal is asynchronously asserted and causes the chip to enter a break sequence, assuming breaks have been previously enabled, and STC2 has been executed.
41	SLRQ+	**Name:** Slave Request **Function:** This input, if enabled at power on by being de-asserted, causes the chip to enter the SLAVE/BREAK mode and to assert PLSLV+, when asserted. **Timing:** At power on, the signal must be de-asserted, if SLAVE/BREAK mode is to be enabled, before the assertion of PON+. Subsequently, the signal may be asserted at any time.
42-48-52-60	CB0+ to CB15+	**Name:** Chip Bus, Bit 0 through Bit 15 **Function:** These signals are bi-directional input/output signals used to input instructions or data into the chip, as inputs, or output control words, addresses, or data, as outputs. **Timing:** At power on the Select Code and Board Identification (SC and ID) of the chip are loaded into the SC and ID Register when PON+ asserts. During an instruction fetch, data is loaded into the Instruction Register Latch (IRL) on the first negative transition of SCLK+ following first positive transition of SCLK+ after VALID+ de-asserts. During the second handshake of an OTA/B data is loaded into the selected register on the third negative transition of SCLK+ following the assertion of IOGO+.

Table 4-5. I/O Processor Chip Signal Definitions (Continued)

PIN NO.	SYMBOL	
		During a DMA self-configuration data is loaded into the appropriate DMA register on the first positive transition of SCLK+ after the de-assertion of VALID+. For I/O handshakes, control words or data from the selected register, if an LI* or MI*, are read out on the first negative transition of SCLK+ following the assertion of IOGO+. During DMA self-configuration the chip outputs successive addresses on the first positive transition of SCLK+ following the assertion of LDMAR+. During DMA transfers the chip outputs Addresses and Word Count Residue, if enabled, on the first positive transition of SCLK+ following the assertion of MRQ+, or the assertion of MRQ+ itself, whichever is later.
61	MEMGO+	**Name:** Memory Go **Function:** This signal is used to de-assert DMACYC− during a DMA transfer or self-configuration and indicates the initiation of a memory cycle. **Timing:** The signal is enabled into a transparent latch while SCLK+ is asserted.
62	MRQ+	**Name:** Memory Request **Function:** This signal is used to assert the first LDMAR+ during DMA self-configuration, LDMAR+, INTCYC+ and the Bus Control Word bits, BCW0+, BCW1+, and BCW2+ during DMA transfers. **Timing:** The signal is enabled into a transparent latch while SCLK+ is asserted.
63		NOT USED

Input Requirements

There are four signals which may be input to the I/O Master by the user interface. Three of these signals, SLRQ, IRQ and LSBYT, if not used, may be left open. A pull-up resistor inside the I/O chip will keep each of these signals high, or non-asserted. The fourth signal, SRQ, if not used, must be tied high through a resistor by the I/O designer.

Note that SRQ and IRQ are referenced to SCLK in the timing specifications. When designing an I/O card on which SRQ and IRQ may be asserted asynchronously, the runt pulse phenomenon is a potential problem. Briefly, this problem occurs when the inputs and clock to a flip-flop change simultaneously, causing the flip-flop outputs to oscillate or to temporarily take on undefined logic levels. In order to minimize the possibility of this occurring, SRQ and IRQ are doubly clocked inside the I/O chip. That is, they are clocked into a first flip-flop on one edge of SCLK. The output of this flip-flop can oscillate without harm until the next clock edge when the data goes through into the second flip-flop, and is acted upon.

In conclusion, if SRQ or IRQ are asserted due to asynchronous events, they may be fed directly into the I/O Master without any external synchronization to SCLK required. LSBYT should always be asserted at least 10 nanoseconds before the assertion of SRQ in order to avoid similar synchronization problems.

DMA Rate Calculation

The maximum achievable DMA rate of an interface is dependent upon the manner in which that interface handshakes with the I/O Master. The equations presented here can be used to calculate the DMA rate for a particular interface. These equations use the simplifying assumption that SACK clears SRQ; that is, that SRQ is never asserted on two consecutive rising edges of SCLK.

First, the interface must be characterized by determining the value of the four following parameters.

a. BYTE
   BYTE = 1 in byte mode
   BYTE = 0 in word mode

b. NRRH (number of cycles to re-request service when LOBYT- is high)
   NRRH is equal to the number of rising edges of SCLK- between two consecutive high-to-low transitions of SRQ- while LOBYT- is high.

c. NRRL (number of cycles to re-request service when LOBYT- is low)
   NRRL is equal to the number of rising edges of SCLK- between two consecutive high-to-low transitions of SRQ- while LOBYT- is low.

d. DIR (Direction)
   DIR = 4 for input
   DIR = 5 for output

Once the value of each of these four variables is determined, they can be substituted into the following equations using the rule that "any expression contained in parentheses that is negative is set equal to zero".

Case 1: For NRRL + BYTE = O we have:
NDMA = 2*

*(= 3 for an A900/A990 memory Read; = 2 for an A900/A990 memory Write)

Case 2: For NRRL + BYTE = > 1 we have:

$$NDMA = DIR + (NRRL - DIR) + (BYTE)(2 + NRRH)$$

Now that NDMA (number of cycles per DMA transfer) is determined, the DMA rate in words per second is given by:

$$\text{DMA rate} = \frac{0.96}{NDMA\ (227X10^{-9})} \quad (A400/A600+)$$

$$\text{DMA rate} = \frac{0.96}{NDMA\ (250X10^{-9})} \quad (A700/A990)$$

$$\text{DMA rate} = \frac{0.96}{NDMA\ (267X10^{-9})} \quad (A900)$$

*This factor varies due to cache hit/miss ratio and memory refresh.

The 0.96 in the numerator accounts for the time the memory is available (4% of memory time is spent refreshing the dynamic RAM chips).

I/O Interface Card Specifications

The required dimension specifications for A-Series interface cards are shown in Figure 4-31. The card shown in Figure 4-31 is a typical Hewlett-Packard A-Series I/O Interface Card shown from the component side of the card. The breadboard card comes with the I/O Master section completely loaded as shown on the assembly drawing in Figure 4-20. A complete I/O Master parts list is located in Table 4-3.

The card is a six layer module printed circuit board (172mm x 289mm). All 62 nodes (53 signals, 8 voltages, and ground) are brought out and labeled, except for the 10 Board ID Code lines on the breadboard I/O card, as shown in Figure 4-21. Ground and + 5V dc are provided on layers 4 and 5, respectively. Signal layers 1,2,3, and 6 may be used to connect to any of the 53 signal nodes and any of the power nodes.

The external I/O connector shown in the upper portion of Figure 4-31 has no standard size or spacing, however, the I/O breadboard card comes with a 48-pin connector. As shown in Figure 4-31, the two Backplane printed circuit connectors, P1 and P2, each consist of 50 conductors. These 100 connections carry power, ground, data, and control signals to and from the I/O card.

Table 4-6. Signal Definitions – I/O Master/User Interface

BCS1– to BCS3– and BCS5– to BCS7–

NAME: Bus Control State 1-3 and 5-7 (low true)

DRIVEN BY: Bus Control State Decoder (U46)

FUNCTION: The Bus Control Signals are used to distinguish between control, status, and data transfers. The Bus Control Signals, except for BCS0 and 4, are user definable. The functions described below are for typical operations:

BCS1: Enables data off the data bus. Asserted in case of:

1. DMA cutput;
2. OTA/B 30 with Global Register enabled and set (GR=SC) for "this" I/O interface;
3. OTA/B SC with Global Register off.

BCS2: Enables control word off data bus. Asserted in case of:

1. OTA/B 31 with Global Register set;
2. DMA self-configuration quadruplet.

BCS3: Latches auxiliary output off the data bus. Asserted in the case of OTA/B 32-77.

BCS5: Enables data onto the data bus. Asserted in case of:

1. DMA input;
2. LIA/B 30 with Global Register set;
3. LIA/B SC with Global Register off.

BCS6: Enables the interface's control word onto the data bus. Asserted in case of LIA/B 31 with Global Register set.

BCS7: Enables the interface's status word or auxiliary input onto the data bus. Asserted in case of LIA/B 32-77.

TIMING: Each state is asserted for at least one cycle of SCLK. States 5, 6, and 7 all prompt the interface to drive data onto the backplane. This data must be valid on the backplane within 70 ns of the assertion edge of the state and must remain valid until the state is deasserted. Latching data off the data bus during states 1, 2, and 3 must be synchronized with the rising (trailing) edge of CKDAT–. Both edges of CKDAT– will occur during these states.

CCLK+

NAME: Communications Clock, high true

DRIVEN BY: Central Processor

FUNCTION: This clock provides a fixed frequency which may be used to drive a state machine or divided down for baud rate generation.

TIMING: CCLK IS A 14.7456 MHz clock with a 50% duty cycle. It is not synchronized to any of the other backplane clocks.

Table 4-6. Signal Definitions - I/O Master/User Interface (Continued)

CFF–	
NAME:	Control Flip-Flop, low true
DRIVEN BY:	IOP chip (U67)
FUNCTION:	This signal informs the user interface as to the state of the control flip-flop. The control FF must be set before the user interface can cause an interrupt to be generated with an IRQ.
TIMING:	Asserted with the execution of an STC 30 and deasserted with the execution of CLC 30 or CLC 0.
CKDAT–	
NAME:	Clock Data, low true
DRIVEN BY:	IOP chip (U67), IOCLK and BVALID are NORed to produce CKDAT–. Indicates that the data bus is about to be latched.
FUNCTION:	Indicates that the data on the bus is about to be valid and should be latched into the user interface data register.
TIMING:	CKDAT– is asserted for one cycle of SCLK. The data bus is valid for a minimum of 50 ns before the trailing edge of CKDAT– and is held 40 ns minimum after the rising edge of CKDAT–.
(DB0+)-(DB15+)	
NAME:	Data Bus 0 - Data Bus 15 , Tri-state, high true
DRIVEN BY:	All system cards
FUNCTION:	DB0 to 15, of which DB0+ is the least significant bit, are used for all system data transfers.
TIMING:	Memory drives the data bus on a read cycle for one cycle, starting one cycle after BUSY– goes low. The central processor drives the data bus with the assertion of MEMGO– on a memory write (STA), with IOGO– on an I/O write (OTA), and with VALID clocked by start of a long half-cycle on A or B fetch or boot read. Refer to BCS1-BCS7, which are the signals indicating to the I/O interface when the data bus is valid and when to drive it.
DMAEN–	
NAME:	DMA Enable, low true
DRIVEN BY:	IOP chip (U67)
FUNCTION:	This signal informs the user interface when DMA is running.
TIMING:	Asserted when an STC 20 or STC 21 is executed. Released with the assertion of the last SACK– on DMA input, and with the penultimate SACK– on DMA output. DMAEN– is also de-asserted when a CLC 0, CLC 22, or CLC 23 is executed, and at power on (PON).

Table 4-6. Signal Definitions – I/O Master/User Interface (Continued)

DVCMD–

NAME: Device Command, low true

DRIVEN BY: IOP chip (U67)

FUNCTION: Signals user interface to perform a data transfer. This signal is generated under program control when an STC 30 to the interface is executed. During DMA, DVCMD– is asserted for each data transfer if the DVCMD (bit 14) in the DMA control word 1 is set.

TIMING: DVCMD– is asserted for one cycle of SCLK.

FETCH–

NAME: Fetch instruction, low true

DRIVEN BY: I/O Master

FUNCTION: When asserted, indicates that an instruction has been fetched from memory and is on the data bus. If other select codes or registers are implemented than those recogized by the IOP, then the user interface may use FETCH to latch the instruction for its own decoding.

TIMING: The data bus should be latched on the trailing, or rising edge of FETCH–.

ID1+-ID10+

NAME: Board Identification Code Bits 1-10, high true

DRIVEN BY: I/O Master

FUNCTION: These 10 bits provide a software accessible board I.D. code. The codes that are presently defined are as follows.

CODE BIT NO. 0987654321	INTERFACE
0000000000	12005A Asynchronous Serial Interface
0000001000	12006A Parallel Interface
1000010000	12007A/12044A HDLC Interface
0000011000	12008A PROM Storage Module
0000100000	12009A HP-IB Interface
0000101000	12010A Breadboard Inteface
1000110000	12040A MUX Interface
0100000000	12060A High Level Analog Input
0100001000	Reserved
0100100000	12062A Digital to Analog
0100010000	Reserved
0100011000	Reserved
1100001000	12063A 16 IN/16 OUT Isolated Digital
1100011000	12072A Factory Data Link Card

NOTE: ID10 is set on intelligent interface cards that have the capability of performing their own self test. When the self test portion of an L-Series diagnostic is executed, the ID10 bit on each card is checked. If this bit is set, the diagnostic will give the interface up to 10 seconds to complete its own self test.

The bits ID1 to ID10 will be placed in bits 6 through 15 of the A/B register when an LIA/B is executed in diagnose mode 1.

TIMING: The ID code bits are hardwired to ground and Vcc when the interface is fabricated.

Table 4-6. Signal Definitions – I/O Master/User Interface (Continued)

IOGO+	
NAME:	I/O Handshake Request Acknowledge, high true
DRIVEN BY:	Central Processor
FUNCTION:	Asserted to signal that the central processor is ready to receive a command or send or receive an operand from an interface card. De-asserted when the transfer has been completed.
TIMING:	Pulled low when the data bus is available for transfers and released as soon as the data has been clocked off the backplane.
NOTE:	For some types of I/O transfers this signal will participate in a double handshake. If this signal occurs during any BCS assertion, then programmed I/O (rather than DMA) is taking place.
IRQ–	
NAME:	Interrupt Request, low true
DRIVEN BY:	Interrupt generation logic, break detection logic, etc. from user interface. Applied as an input to the IOP chip (U67).
FUNCTION:	Asserted to set FLG 30 in the IOP to indicate to the CPU that an interrupt condition exists.
TIMING:	Clocked into the IOP chip on ↑SCLK–, but will cause an interrupt only if CFF is asserted.
LDCW1–	
NAME:	Load Control Word 1, low true
DRIVEN BY:	I/O Master
FUNCTION:	This signal is asserted to indicate that DMA control word 1 is valid on the backplane.
TIMING:	The data bus should be latched on the trailing edge (rising edge) of LDCW1–.
LOBYT–	
NAME:	Low Byte, low true
DRIVEN BY:	IOP chip
FUNCTION:	This signal is used to implement byte mode DMA, where bytes are packed into 16-bit words. When LOBYT– = 0, the data is routed to the lower byte of the user interface data register. When LOBYT– = 1, the data is routed into the upper byte of the data register. LOBYT is always low when DMA is not enabled, and when word mode DMA is running.
TIMING:	LOBYT– switches to the appropriate sense within 30 ns after the assertion of SACK–.
LSBYT–	
NAME:	Last Byte, low true
DRIVEN BY:	User Interface (Special character recognition logic)
FUNCTION:	Provides a method of terminating a DMA block transfer before a word/byte count rollover occurs. Sets FLG 21, or in the case of self-configured DMA with the continue bit set (CONT bit of register 21), the IOP will automatically configure for the next transfer.
TIMING:	Must be asserted before the leading edge of the SRQ– which requests the last data transfer.

Table 4-6. Signal Definitions – I/O Master/User Interface (Continued)

MRQ+	
NAME:	Memory Request, high true
DRIVEN BY:	I/O Master MRQ FF (U86A)
FUNCTION:	MRQ is asserted when the I/O Master requests a memory cycle in order to do a DMA transfer. MRQ is inverted (MRQ–) to the backplane to inhibit the central processor from requesting a memory cycle.
TIMING:	MRQ is asserted at the start of the long half cycle of SCLK. It is released a minimum of two states later, once the memory cycle has been granted.
PON+/–	
NAME:	Power On, both high and low true available
DRIVEN BY:	Power Supply
FUNCTION:	Asserted to indicate that all power supplies are stable. Deasserted if any power supply falls below a tolerable level.
TIMING:	Asserted 50 ms after all supplies are stable.
RST–	
NAME:	Reset, low true
DRIVEN BY:	Power supply and central processor
FUNCTION:	This signal may be used to initialize logic circuits. It is asserted by the power supply during power up and released 50 ms after all power supplies are stable. Under program control, RST– is asserted upon execution of a CLC 0.
TIMING:	Asserted for one cycle of SCLK.
SACK–	
NAME:	Service Request Acknowledge, low true
DRIVEN BY:	IOP chip (U67)
FUNCTION:	The assertion edge of SACK– acknowledges a Service Request (SRQ–). It is not released until after SRQ– is released. When released, it indicates that the IOP is ready to service a new SRQ. Under program control, SACK– will not be released until FLG 30 is programmatically cleared.
TIMING:	Asserted on the rising edge of SCLK if SRQ– was present on the previous clock edge, and if the IOP does not have more than two previous SRQ's still waiting to be serviced.

Table 4-6. Signal Definitions – I/O Master/User Interface (Continued)

SCLK+	
NAME:	Slow Clock, high true
DRIVEN BY:	Central Processor
FUNCTION:	SCLK is used to synchronize many diverse system signal interactions.
TIMING:	SCLK is a derivative of FCLK. It is generated with a divide-by-5 circuit which produces a signal with a minimum period of 227.1ns (250ns in A700) and a 40% duty cycle.
NOTE:	In all timing descriptions, the term "short half cycle" refers to the time (2/5 period) when SCLK– is high (SCLK+ is low). The term "long half cycle" refers to the 3/5 period when SCLK– is low (SCLK+ is high). To minimize clock skew, all cards are required to receive SCLK into an S240 IC.
SLAK–	
NAME:	Slave Request Acknowledge, low true
DRIVEN BY:	I/O Master
FUNCTION:	SLAK– acknowledges the receipt of a slave request. It may be used as a means of clearing the slave request (SLRQ).
TIMING:	SLAK is an asynchronous signal. It is asserted as soon as the IOP receives SLRQ, and released as soon as the central processor enters slave mode.
SLRQ+	
NAME:	Slave Request, high true
DRIVEN BY:	User Interface
FUNCTION:	The sense of SLRQ on the rising edge of PON tells the IOP whether the interface ever wants the capability of requesting slave mode. If SLRQ is high at PON, then any subsequent change of this line will be ignored, and slave mode will never be invoked by this card. If SLRQ is low at PON, then the IOP will put the central processor in slave mode on any subsequent assertions of SLRQ. When in slave mode, the IOP essentially jumps into the Virtual Control Panel Program (VCPP) after saving the current CPU P-register value.
TIMING:	SLRQ+ is asserted to request slave mode, and should be held asserted at least until the long half cycle of SCLK following the release of SCHID–. SLRQ should be deasserted sometime before the VCPP has finished executing.
SRQ–	
NAME:	Service Request, low true
DRIVEN BY:	User Interface
FUNCTION:	Initiates a data transfer. Under program control, the assertion of SRQ– causes Flag 30 to be set. During DMA, in word mode, SRQ– causes the IOP to initiate (steal) a memory cycle. During a byte mode DMA, only every other SRQ– will cause a memory access, and the odd SRQ's will only affect the byte packing signals.

Table 4-6. Signal Definitions – I/O Master/User Interface (Continued)

TIMING:	The IOP is ready to receive an SRQ anytime that SACK– is not asserted. SRQ– is clocked into the IOP on the falling edge of SCLK+ and acknowledged on the next edge of that clock. SRQ can be released anytime after SACK– is asserted. In the case of self-configured DMA input, if SRQ is held low on the last data byte, the IOC will hold off configuration until SRQ– is released.
SRQM+	
NAME:	Service Request Mask, high true
DRIVEN BY:	I/O Master
FUNCTION:	When this signal is asserted, the I/O Master will ignore any assertions of SRQ–.
TIMING:	Asserted upon execution of an OTA 2 with a 7 in the A-register (places the I/O Master in diagnose mode 7, which allows diagnostics to be executed). Deasserted on power-up or on execution of a CLC 0.

Table 4-7. Definitions of Terms Used in Timing Diagrams and Timing Specifications

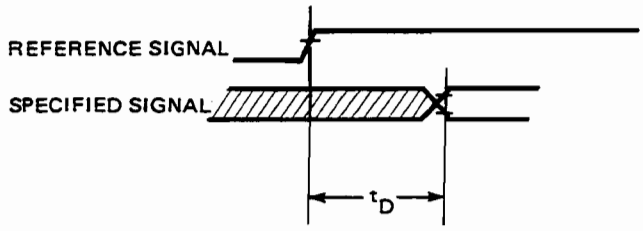
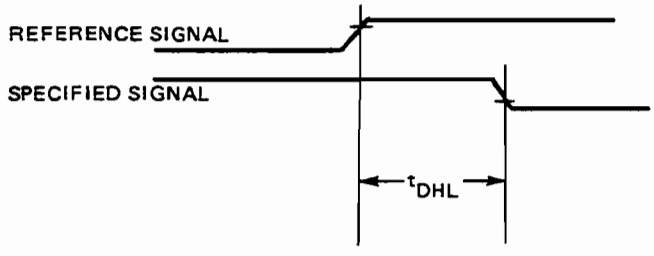
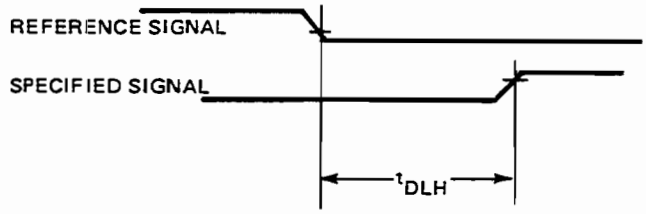
C	Cycle One cycle of slow clock
f	Frequency The number of cycles per unit time of a given signal.
LHC	Long Half Cycle The Long Half Cycle refers to the time period when SCLK– is low.
SHC	Short Half Cycle The Short Half Cycle refers to the time period when SCLK– is high.
t_D	Delay Time The time interval from a signal edge used as a reference point to the point in time when the specified signal is guaranteed to be stable at the user interface. 
t_{DHL}	Delay time high to low The time interval from a signal edge used as a reference time to the point in time when the specified signal is guaranteed to be low if in fact it is going low. 

Table 4-7. Definitions of Terms Used in Timing Diagrams and Timing Specifications (Continued)

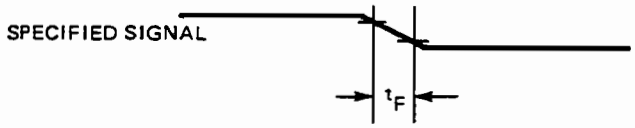
t_{DLH} Delay time low to high

The time interval from a signal edge used as a reference time to the point in time when the specified signal is guaranteed to be high if in fact it is going high.



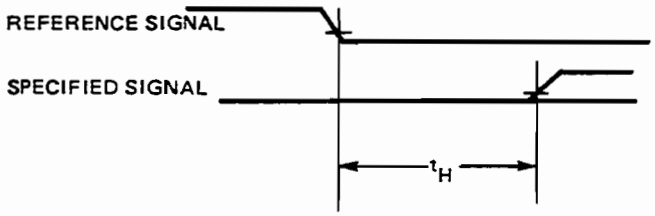
t_F Fall time

The time interval when a signal is in transition from high to low.



t_H Hold time

The period of time during which a specified signal must remain stable at its logic level after a certain reference edge.



t_p Period

The duration of one cycle of a periodic signal.

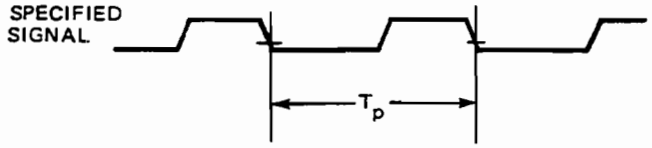
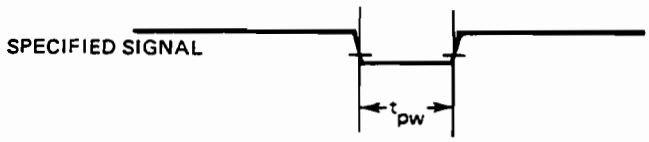


Table 4-7. Definitions of Terms Used in Timing Diagrams and Timing Specifications (Continued)

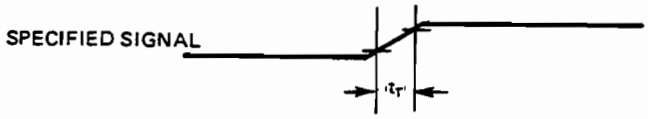
t_{pw} Pulse width time

The time interval btween the leading and trailing edge of a pulse. Specifically, for a normally high signal, t_{pw} is the time when that signal is low. For a normally low signal, t_{pw} is the time when that signal is high.



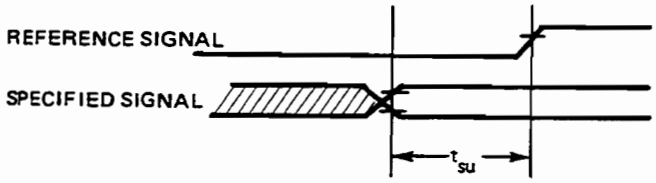
t_r Rise time

The time interval during which a signal is in transition from low to high.



t_{su} Set-up time

The time interval a specified signal must be at a stable logic level before a given edge of a reference signal.



NOTE

In these timing digrams a high notch is 2.0 Volts and a low notch is 0.4 Volts as shown below.

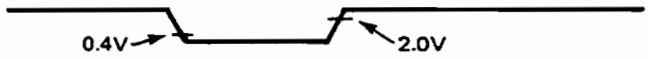


Table 4-8. BCS- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{DHL}	SCLK-↓	During I/O handshake, ref. is First SCLK-↓after second IOGO-↓			170
t_{DLH}	SCLK-↓	During I/O handshake, ref. is 1C after above	60		170
t_D	SCLK-↓	During DMA, ref. is same edge that causes MEMGO-↓			45
t_{DLH}	SCLK-↓	During DMA, ref. is first SCLK-↓after VALID-↑ (BCS 1 and 2 only)	60		180

Table 4-9. CCLK- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
f (A-Series)	Asynchronous	To all other Backplane Signals	14.7452 MHz	14.7456 MHz	14.7460 MHz
Duty Cycle			30%	50%	

Table 4-10. CFF- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
T_D	SCLK-↑	Set and cleared upon execution of STC and CLC instructions respectively			135

Table 4-11. CKDAT- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
T_D .	SCLK-↓	During programmed I/O, ref. to subsequent SCLK edges during 2nd IOGO			140
t_D	SCLK-↑	During DMA, ref. to the edge of SCLK that causes VALID-↓↑			65
T_{PW}			200	1C	300

Table 4-12. DB0–DB15* Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{SU}	FETCH-↑	Backplane to I/O card	50		
t_H	FETCH-↑	Backplane to I/O card	40		
t_{SU}	CKDAT-↑	Backplane to I/O card	50		
t_H	CKDAT-↑	Backplane to I/O card	40		
t_{SU}	LDCW1-↑	Backplane to I/O card	50		
t_H	LDCW1-↑	Backplane to I/O card	30		
* The DATA BUS is driven directly onto the backplane. The 16 data bus lines are the only lines other than power available to the I/O Master user which are not buffered by the I/O Master circuitry.					
t_D	BCS-↓	I/O card to backplane			125
t_H	BCS-↑	I/O card to backplane	20		

Table 4-13. DMAEN– Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{DLH}	SCLK-↓	Referenced to first SCLK- | that occurs during n^{th} SRQ			140

Table 4-14. DVCMD– Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_D	SCLK-↓				90
t_{pw}			150	1C	300
t_{DLH}	SACK-↓	Output only	5C		12C
t_{DLH}	SACK-↓	Input only	3C		10C

Table 4-15. FETCH– Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{PU}			200	1C	270
t_D	VALID-↓↑				13

Table 4-16. IRQ- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{SU}	SCLK-↑	IRQ-	0		
t_H	SCLK-↑	IRQ-	25		

Table 4-17. LDCW1- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_D	SCLK-↓	During programmed I/O, ref. to subsequent SCLK edges during 2nd IOGO			150
t_D	SCLK-↑	During DMA, ref. to the edge of SCLK that causes VALID ↓↑			75
t_{PW}			200	1C	250

Table 4-18. LOBYT- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_D	SCLK-↓	Referenced to edge that causes SACK-↓			120
t_D	SCLK-↓	Referenced to edge that causes DMAEN-↓			120

Table 4-19. LSBYT- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{SU}	SCLK-↑	LSBYT- Referenced to same edge as SRQ-	0		
t_H	SCLK-↑	LSBYT- Referenced to same edge as SRQ-	25		

Table 4-20. MRQ+ Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{DHL}	SCLK-↓	Referenced to same edge that causes MEMGO-↑			30

Table 4-21. PON+ Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_D		Power supplies are up and within regulation	50 ms	65 ms	100 ms

Table 4-22. RST- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{PU}			180	1C	
t_D	SCLK-↑				70

Table 4-23. SACK- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_D	SCLK-↓				80
t_{DHL}	SRQ-↓	Provided SRQM not asserted	100		
t_D	DMAEN-↓	If AUTO bit set on DMA output	90		300
t_{PW}			200	1C	300

Table 4-24. SCLK+ Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
f	A400/A600+		-0.005%	4.403 MHz	+0.005%
Duty Cycle	A400/A600+		33%	40%	47%
f	A700/A990		-0.01%	4.400 MHz	+0.01%
Duty Cycle	A700/A990		33%	40%	47%
f	A900		-0.01%	3.70 MHz	+0.01%
Duty Cycle	A900		43%	50%	57%

Table 4-25. SLRQ+ Timing Specifications

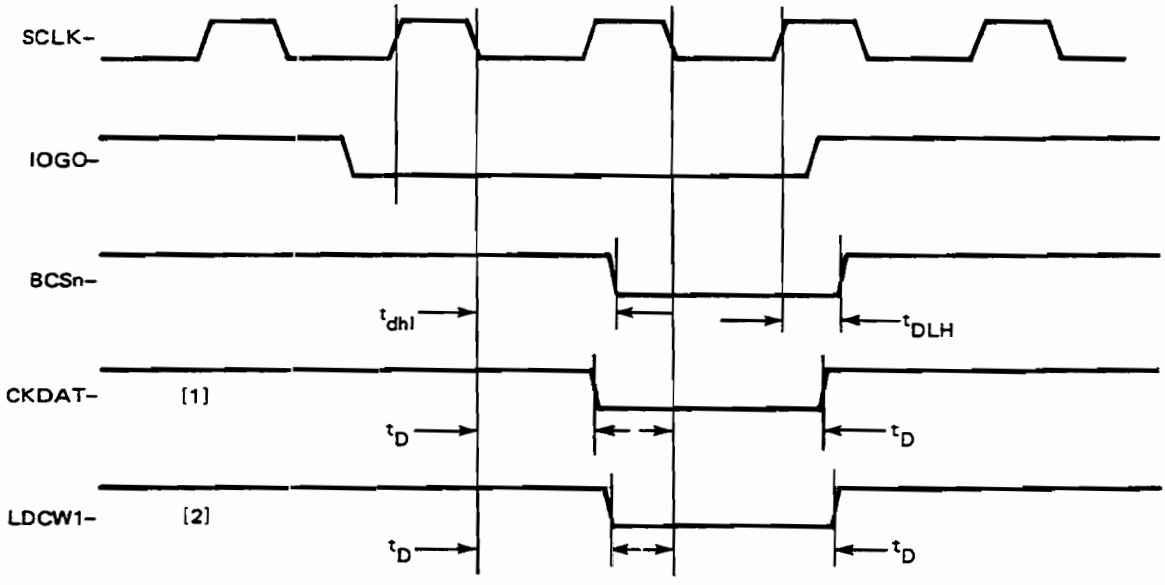
PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{SU}	PON+	SLRQ+	25		
t_H	PON+	SLRQ+	25		
t_{PW}	PON+	SLRQ+	50		

Table 4-26. SRQ- Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{SU}	SCLK-↑	SRQ-	10		
t_H	SCLK-↑	SRQ-	25		

Table 4-27. SRQM+ Timing Specifications

PARAMETER	REFERENCE	NOTES	MIN	TYP	MAX
t_{DLH}	SCLK-↓				165
t_{DHL}	SCLK-↑	Caused by RST-↓			80



[1] CKDAT (AS GENERATED FROM IOCLK) OCCURS ONLY DURING THE SECOND HANDSHAKE OF AN OTA/B INSTRUCTION.

[2] LDCW1 OCCURS ONLY DURING THE SECOND HANDSHAKE OF AN OTA/B21 (SHOWN HERE) AND DURING FOUR WORD SELF CONFIGURATION (SHOWN IN FIGURE 4-23).

Figure 4-21. BCS Timing – I/O Handshake

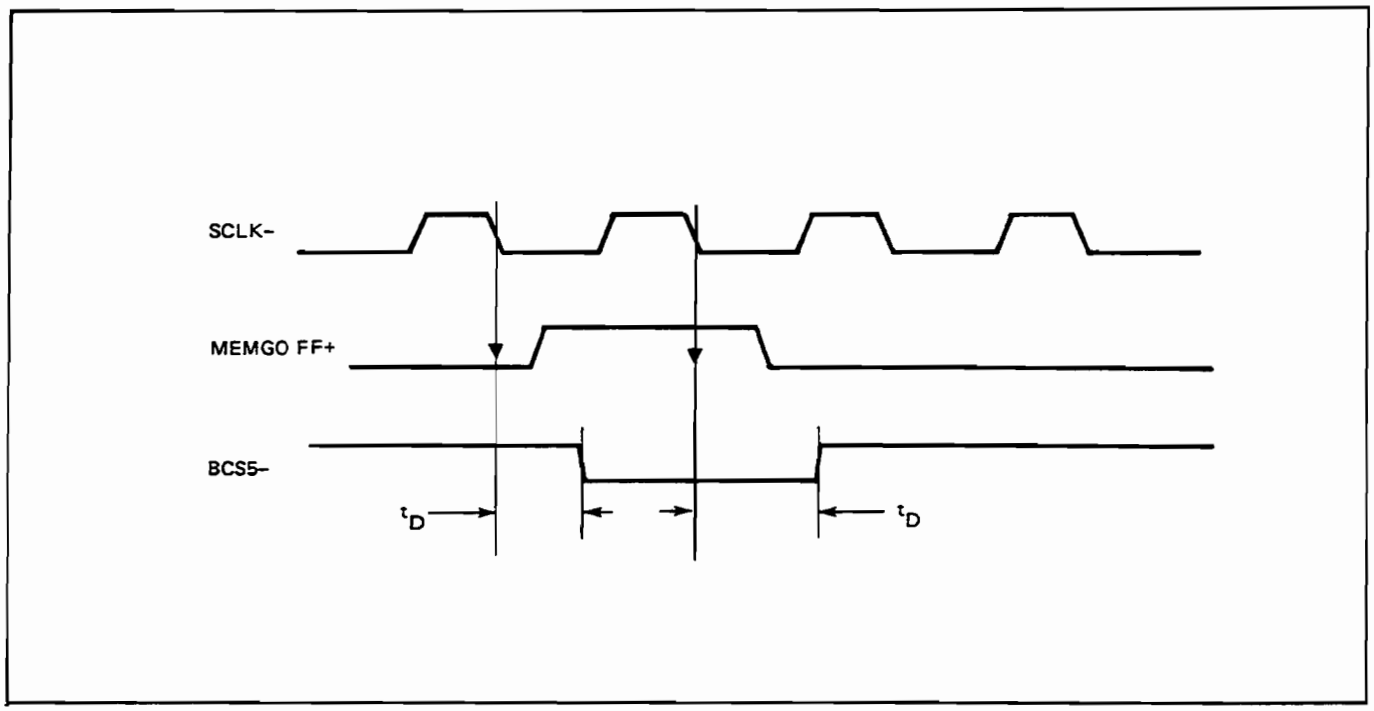


Figure 4-22. BCS Timing – DMA input

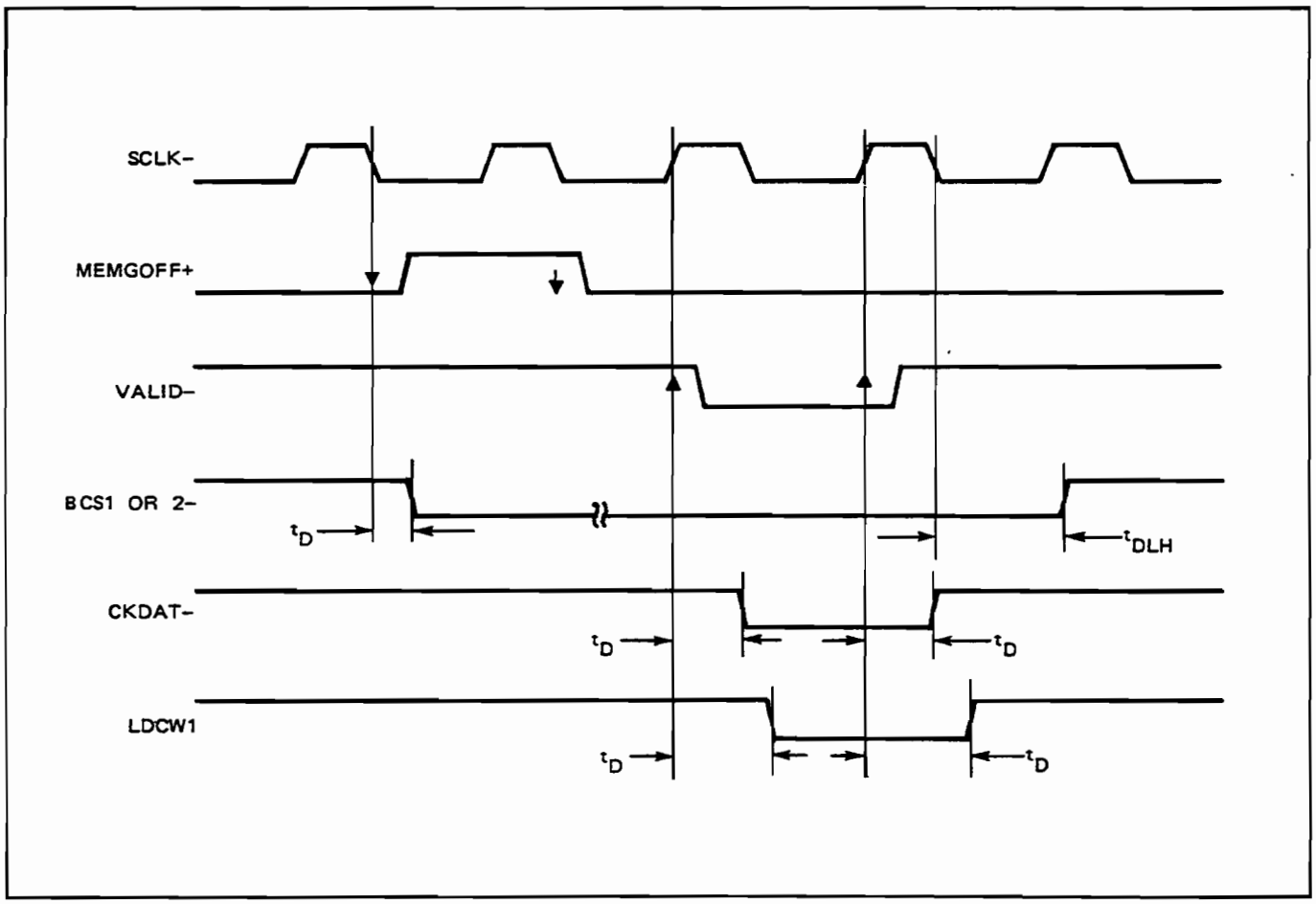
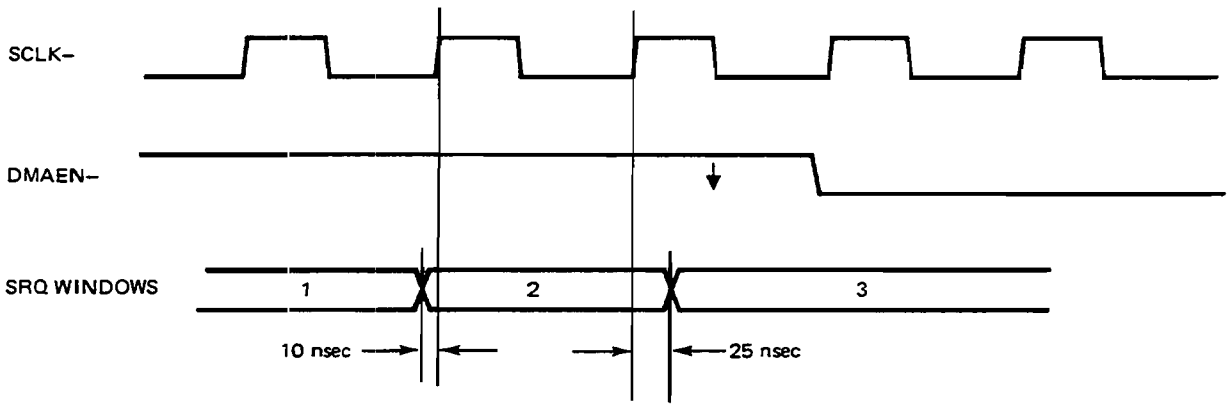
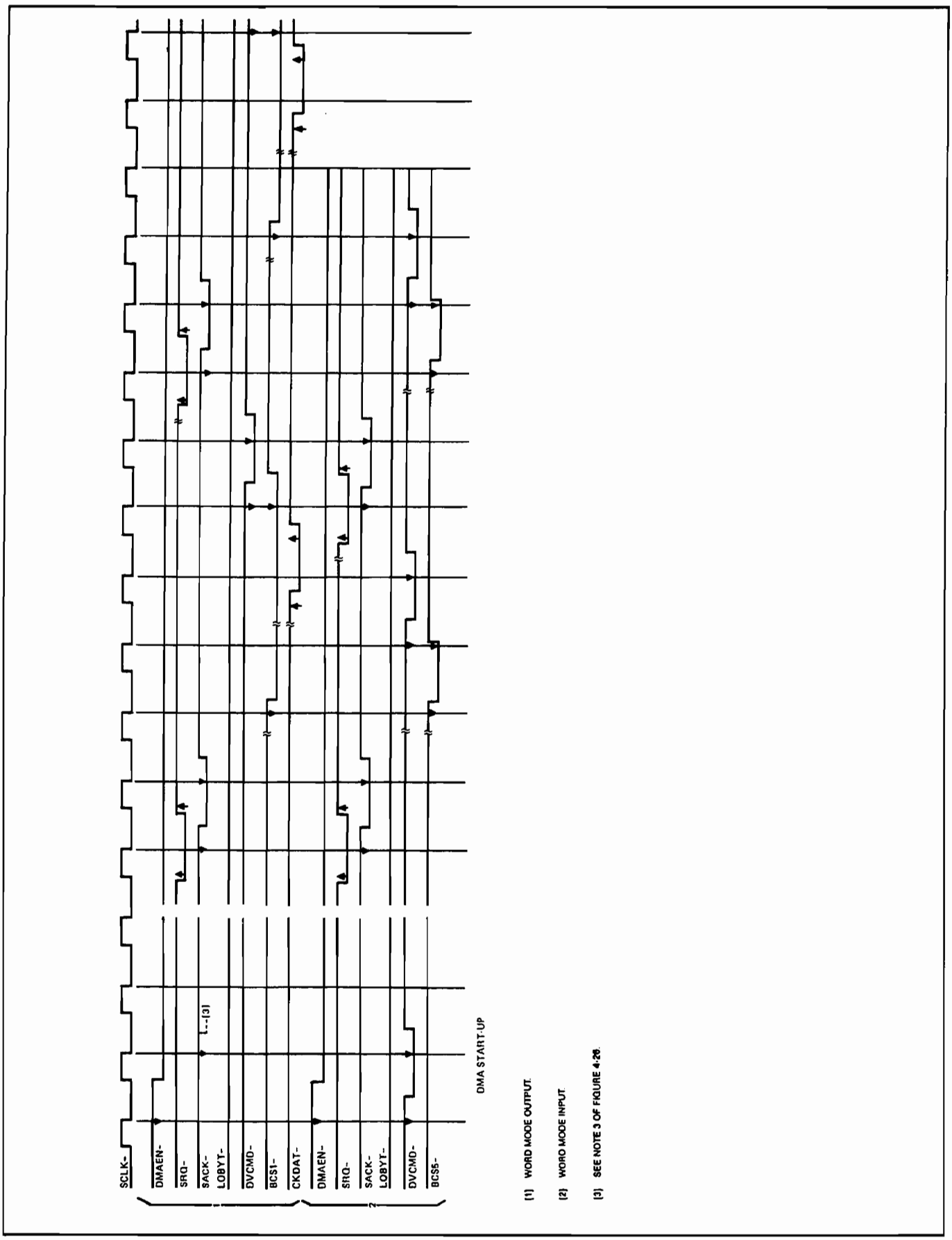


Figure 4-23. BCS Timing – DMA Output and Self-Configuration



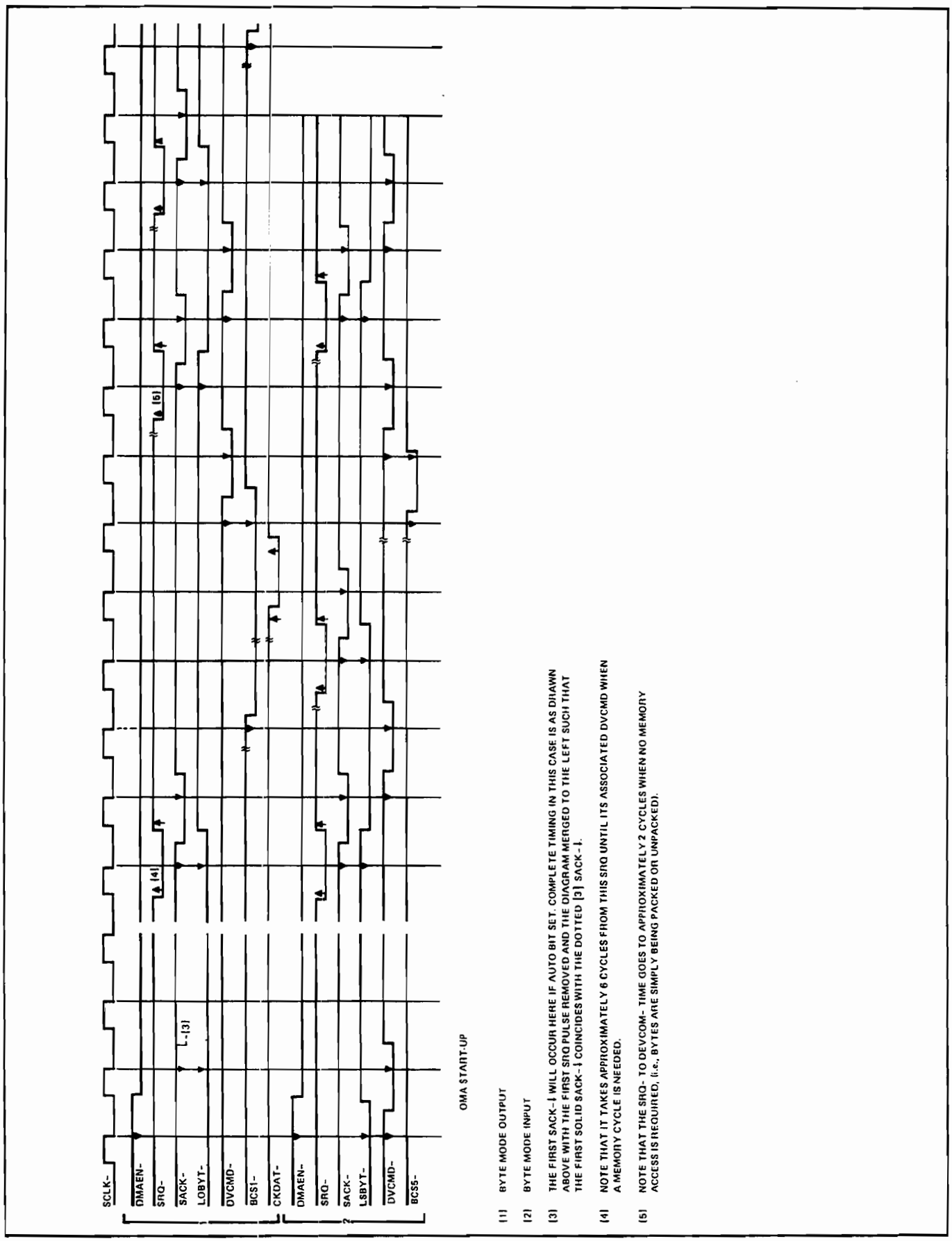
1. AN SRQ– WHICH BEGINS IN THIS WINDOW WILL CAUSE FLAG 30 TO GET SET.
2. AN SRQ– WHICH BEGINS IN THIS WINDOW WILL HAVE UNDETERMINATE RESULTS. POSSIBLY, IT COULD BOTH SET THE FLAG AND START A DMA TRANSFER.
3. AN SRQ WHICH BEGINS IN THIS WINDOW WILL CAUSE A DMA TRANSFER.

Figure 4-24. SRQing on DMA Start-Up



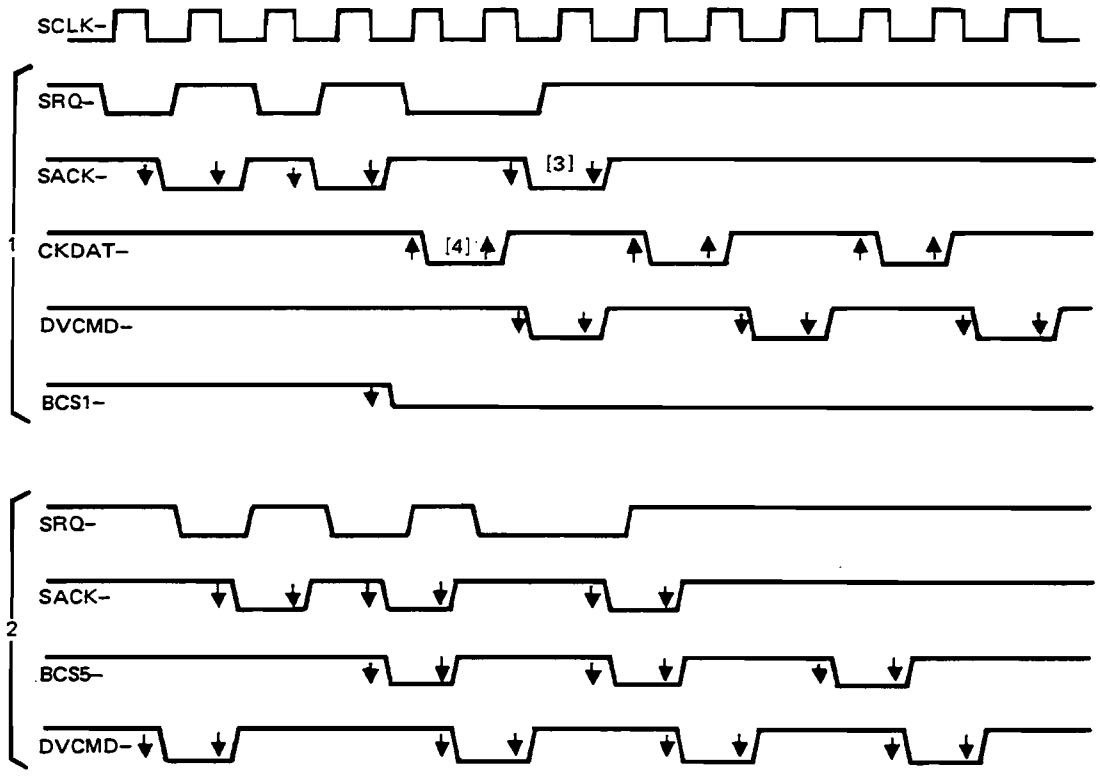
7700-527

Figure 4-25. Low-Speed DMA Transfers – Word Mode



7700-526

Figure 4-26. Low-Speed DMA Transfers – Byte Mode



FULL SPEED DMA IN PROGRESS

[1] WORD MODE OUTPUT

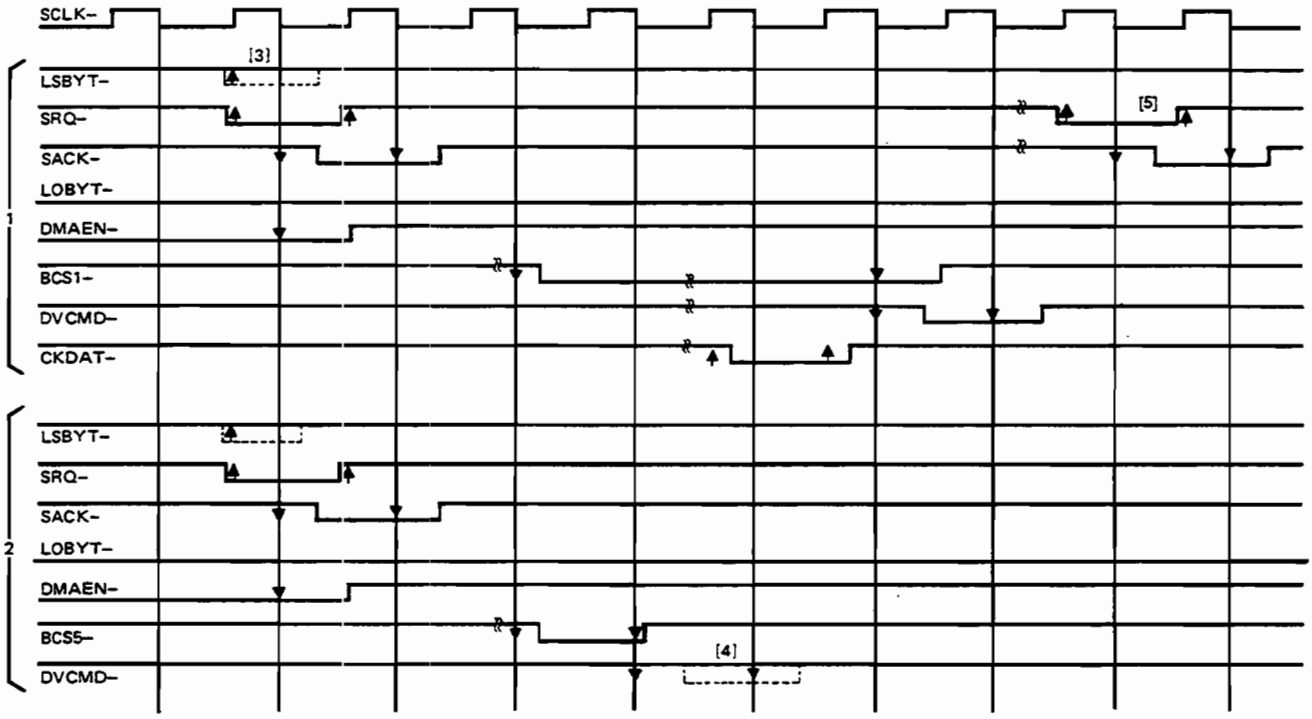
[2] WORD MODE INPUT

[3] NOTE THAT THE THIRD SACK IS DELAYED ONE CYCLE WAITING FOR THE FIRST SRQ TO BE SERVICED.

[4] NOTE THAT THESE PULSES OCCUR EVERY THREE CYCLES. THIS IS DUE TO OUR THREE CYCLE MEMORY, ACCESSED AT EVERY POSSIBLE OPPORTUNITY.

GENERAL NOTE: FULL SPEED DMA IS ONLY POSSIBLE IN WORD MODE.

Figure 4-27. Full Speed DMA In Progress



[1] WORD MODE OUTPUT.

[2] WORD MODE INPUT.

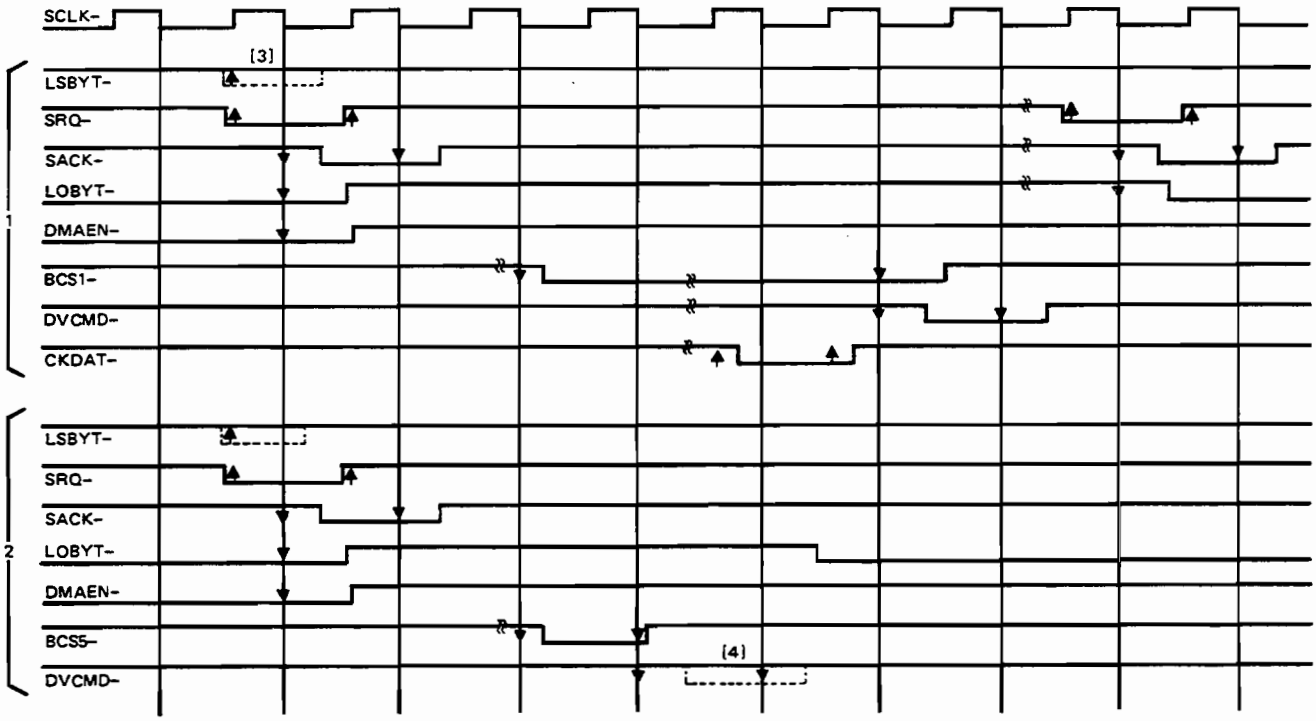
[3] TRANSFERS MAY BE TERMINATED BY WORD COUNT ROLL-OVER OR THE ASSERTION OF LSBYT.

[4] THIS PULSE IN DOTTED LINES WILL OCCUR IF AUTO BIT IS SET.

[5] THIS IS A FINAL SRQ NOT ASSOCIATED WITH A WORD (THAT IS, 100 WORDS TRANSFERRED, 101 SRQs REQUIRED). THIS SRQ IS REQUIRED TO SET FLAG 21 TO NOTIFY THE PROGRAM OF DMA COMPLETION.

7700-529

Figure 4-28. DMA Termination – Word Mode



ODD NUMBER OF BYTES (LAST BYTE IS UPPER BYTE)

[1] BYTE MODE OUTPUT

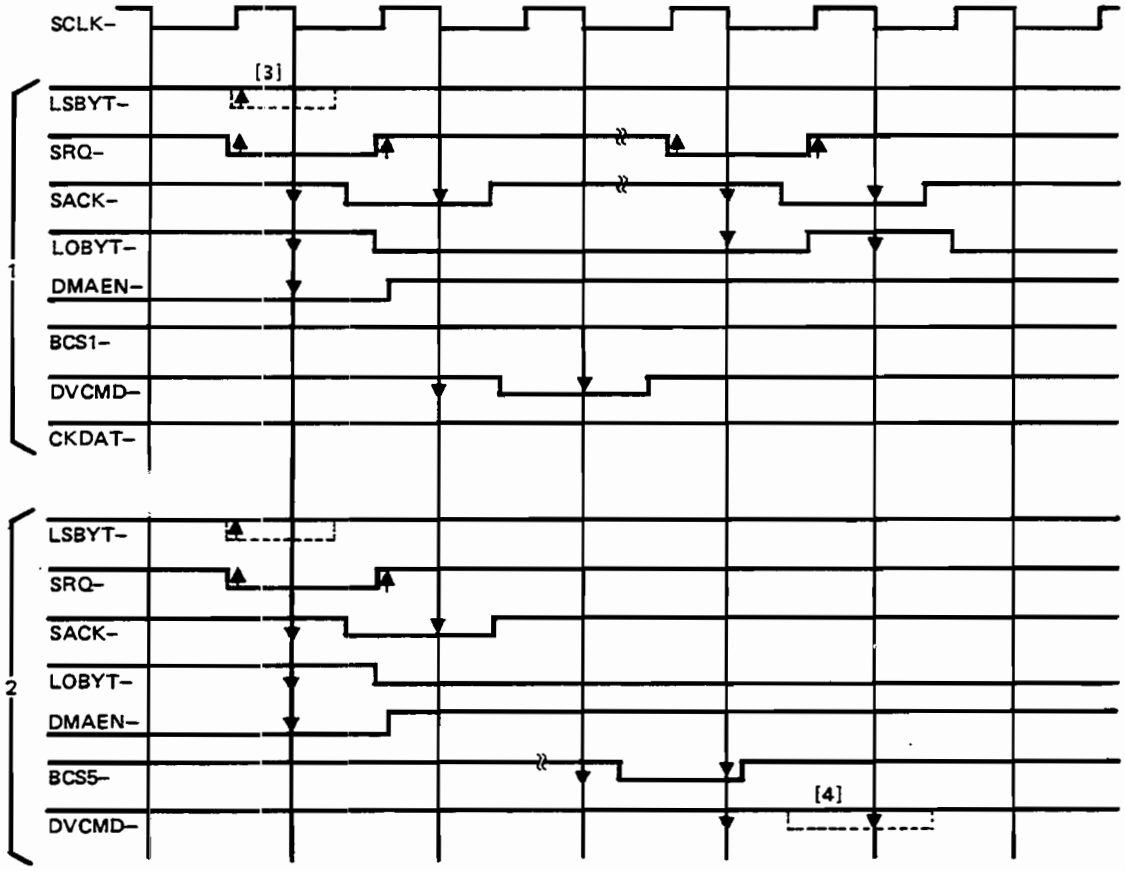
[2] BYTE MODE INPUT

[3] TRANSFERS MAY BE TERMINATED BY WORD COUNT ROLL-OVER OR THE ASSERTION OF LSBYT

[4] THIS PULSE IN DOTTED LINES WILL OCCUR IF AUTO BIT IS SET.

7700-528

Figure 4-29. DMA Termination – Byte Mode (Sheet 1 of 2)



EVEN NUMBER OF BYTES (LAST BYTE IS LOWER BYTE)

Figure 4-29. DMA Termination – Byte Mode (Sheet 2 of 2)

Capacitive Loading

The breadboard trace capacitances for 0.015-inch width traces (nominal), worst case figures are:

Layer 1	2.69 pF/inch
Layer 2	3.66 pF/inch
Layer 3	4.34 pF/inch
Layer 6	4.34 pF/inch

Data bus lines are very sensitive to capacitive loading. Allowable capacitance is limited to 60 pF/line/card to insure overall system operation. Some of this capacitance is taken up by the I/O Master. Use the following table for available capacitance for each data bus line.

SIGNAL	I/O MASTER TRACE	I/O MASTER IC PACK	USER AVAILABLE
DB0	12.4	11.3	36.3
DB1	14.1	11.3	34.1
DB2	12.9	11.5	35.6
DB3	15.6	12.1	32.3
DB4	18.9	12.4	28.7
DB5	23.0	11.4	25.6
DB6	20.3	11.3	28.4
DB7	22.3	12.7	25.0
DB8	17.1	11.3	31.6
DB9	19.7	11.8	28.5
DB10	14.2	11.5	34.3
DB11	21.2	12.1	26.8
DB12	15.1	12.4	32.5
DB13	20.3	11.7	28.0
DB14	16.0	11.3	32.7
DB15	20.0	12.7	27.3

Initial Design Considerations

The initial design considerations include formulating a list of the functions that must be present on the interface. To make up this list, consider questions such as the following:

a. What kind of data registers are required? Will the register be used for output only (to the I/O device), input only (from the I/O device), or will two or more registers be required to handle both input and output operations? How many flip-flops will be required to store all the bits? Are any registers required at all? This may be the case if, for example, the I/O device has its own storage facilities.

b. What commands are required? Are other commands such as tape rewind, upper/lower-case shift, mode switching, etc., required? If so, a command register may be needed to accept command words from the computer. The reverse situation of the computer being slaved to or commanded by the I/O device is also possible. In this case, an input command register may be required. Perhaps no control lines at all are required for the I/O device. On input, for example, a computer program may simply require the current value of a count-accumulating I/O device. The computer need not command the I/O device to read, and the I/O device need not have to inform the computer that data is ready for transfer. Conversely, on output, data may simply be presented to the I/O device without any accompanying commands. For example, this would be possible if the I/O device was a display unit or a device interlocked with some other program-synchronized device (for example, a scanner-voltmeter relationship). In most cases, the recommended approach is to have the computer and I/O device completely interlocked so that each knows exactly what the other is doing.

c. What status information is required? It is often necessary to interrogate a device or interface card as to status, in order to obtain such information as the cause of an interrupt, the state of a control circuit, modem line status, the validity of data, etc. A separate status register may be needed, or, with some devices, status and control can be accomplished with a multiplexer.

d. What type of logic is to be used? The 74LS and 74S TTL logic families are recommended for use in driving the I/O Master because of their high switching speeds, good noise immunity, and compatibility with the other logic circuits on the card.

Development Checklist

The following paragraphs outline the sequence of steps that should be followed when developing an I/O interface card.

Formulate a list of required functions for your interface card.

Draw initial logic diagram. The I/O Master section must be used to interface with the A-Series backplane. Therefore, the first step would be to include the I/O Master section logic diagram from the foldout diagram in the rear of this manual. To complete the diagram, add your new circuits.

Fabricate a model. Depending on the number of I/O cards to be produced, this model will be either the final interface or a prototype. The HP 12010A Breadboard I/O Card should be used to save time in layout and fabrication. The I/O Master section is already in place and the interconnection of the I/O Device Interface completes the design. Contact the nearest HP Sales and Service Office for information on ordering the Breadboard Card.

Testing your interface. Initially, check the card circuit paths for shorts with an ohmmeter. Next, set the select code switches to an available select code and install the interface card into the desired priority A-Series backplane slot using an extender card. Connect the associated I/O device or test fixture to the interface card. Apply power to the computer and I/O device or test fixture. Prepare a simple noninterrupt program to operate the I/O device and checkout the card. For example, you could write a program to read and write from various registers and then to the device. The Diagnostic Design Language (DDL) may be helpful in this phase. DDL is a BASIC-like language intended for preparing diagnostic or troubleshooting programs and execute independent of the operating system. The DDL manual or various I/O diagnostic manuals could give you ideas on what to test.

If the device and interface card work properly in the non-interrupt mode, the next step is to check for proper operation in the interrupt mode. Write a program that checks the interrupt capabilities of the interface card. If DMA is to be used, write a program to check DMA operation.

After the interface card design has been debugged, an RTE interface driver (and a device driver, if needed) should be written to check for proper operation under control of the RTE operating system.

Final test and production. Perform final checkout of the working model under all environmental conditions, and if required, update the logic diagram and layout drawing. If additional cards are to be produced from the working model, prepare a comprehensive production test program to test each card with the computer and I/O device. Completely check each card for marginal signals.

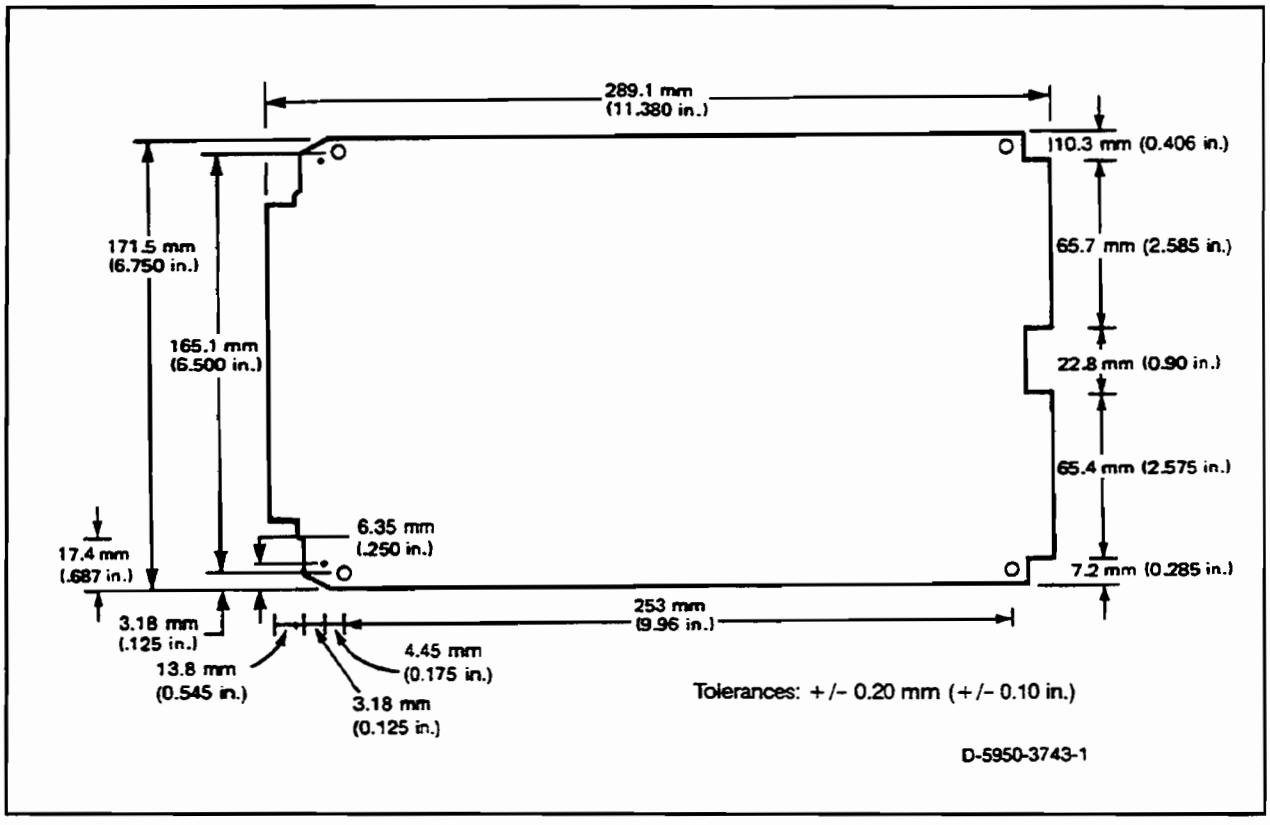


Figure 4-30. A-Series I/O Card Dimensions