Twenty Dollars

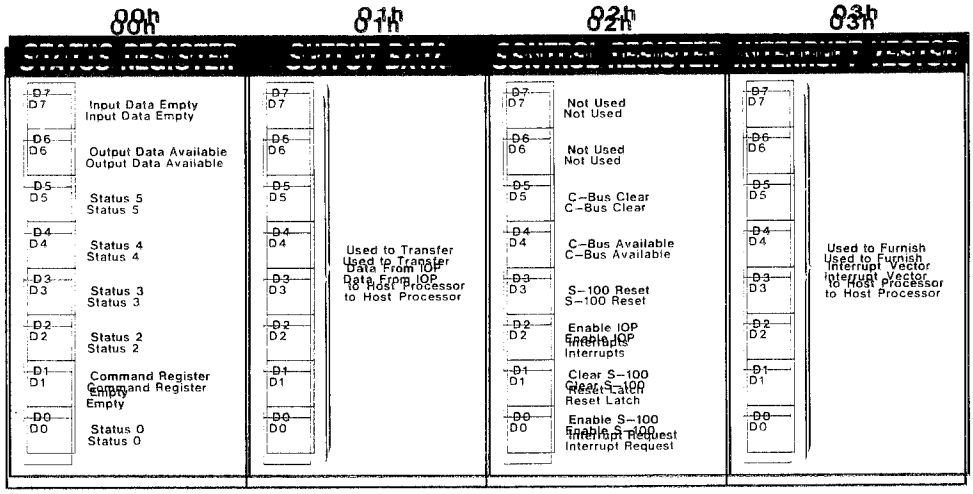
Cromemco

IOP

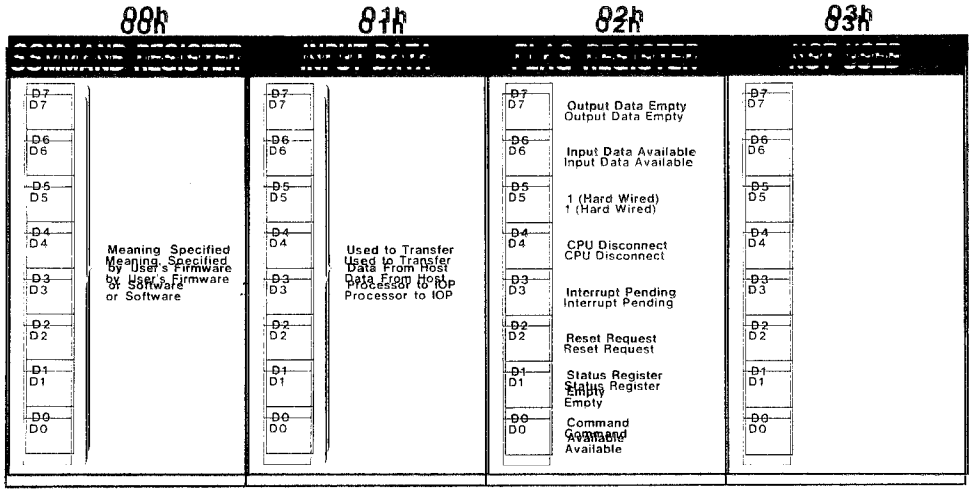
Input/Output Processor

Instruction Manual

IOP OUTPUT PORTS



IOP INPUT PORTS



Cromemco®

Input/Output Processor
C-Bus Controller
(IOP)

Instruction Manual

CROMEMCO, Inc.
280 Bernardo Avenue
Mountain View, CA. 94043

Part No. 023-2006

May 1982

This manual was produced on a Cromemco System Three computer utilizing a Cromemco HDD-22 Hard Disk Storage System running under the Cromemco Cromix™ Operating System. The text was edited with the Cromemco Cromix Screen Editor. The edited text was proofread using the Cromemco SpellMaster™ Program and then formatted using the Cromemco Word Processing System Formatter II. Final camera-ready copy was printed on a Cromemco 3355A printer.

TABLE OF CONTENTS

FIGURES

TABLES

TECHNICAL SPECIFICATIONS

Input/Output Processor

Processor Type:	Z-80A
Clock Frequency:	4.000 MHz
Instruction Set:	158 instructions including the 78 member 8080 instruction set
ROM Capacity:	Up to 32 Kbytes
ROM Type:	Texas Instruments 2516, 2532 EPROM; Texas Instruments 4732, 4764 masked ROM; Intel 2716, 2732 EPROM, or their generic equivalents
RAM Capacity:	16 Kbytes
RAM Type:	4116 Dynamic (included)
IOP Registers:	All registers I/O mapped; 4 output (Status, Output Data, Control and Interrupt Vector), 3 input (Command, Input Data and Flags)
S-100 I/O Ports:	2 output (Command and Input Data), 2 input (Status and Output Data)
C-Bus Interrupts:	Host read/write interrupts and C-Bus peripheral interrupts daisy chain prioritized; Z-80A interrupt modes IM0, IM1 and IM2 supported
S-100 Bus Interrupts:	IOP interrupt requests to host software controlled; IOP supplies programmable IM0 or IM2 vectors; IOP interrupt requests S-100 daisy chain prioritized
Bus Compatibility:	Standard-100 (S-100) Cromemco Bus (C-Bus)
Power Requirements:	+8 VDC @ 1.75 Amps (max) +18 VDC @ 100 mA (max) -18 VDC @ 50 mA (max)
Operating Environment:	0 - 55 degrees Celsius

ABOUT THIS MANUAL

After reading this manual, the reader should be able to install, configure, and verify the proper operation of an IOP board in a host S-100 system. Also, the reader should be able to tailor the IOP configuration options to best suit the system requirements at hand. While this manual provides some IOP programming examples, it is not intended to be a software reference.

Chapter 1 explains Cromemco's C-Bus architecture, and the role the IOP board plays in it. Chapter 2 discusses most IOP hardware features excluding interrupts and DMA operation. An IOP initial checkout procedure appears at the end of Chapter 2. All categories of IOP interrupts are treated in Chapter 3, including interrupt programming examples. Chapter 3 also deals with C-Bus DMA. Chapter 4 analyzes the major IOP hardware subsystems and major signal paths, and it presents C-Bus waveforms for every processor cycle type.

The reader of this manual is assumed to have some knowledge of the Z-80A microprocessor and Z-80 Assembly Language. Throughout the entire manual, positive logic is assumed; **reset** means logic 0, and **set** means logic 1 as these terms apply to bit states.

The following two references should prove helpful while reading this manual. The first, which is included in Cromemco's **IOP Development Software** package, is the primary IOP software reference. It outlines the entire IOP software development cycle, and describes how to operate the IOP utility programs contained in the package. The second describes Cromemco's **Quadart** board, a powerful four channel serial C-Bus peripheral. Copies may be obtained at your Cromemco dealer.

1. Cromemco IOP Development Software Instruction Manual, part number 023-4032

2. Cromemco Quadart Serial Interface Instruction Manual, part number 023-2005

Chapter 1

INTRODUCTION

The architecture of S-100 systems hasn't changed much since the S-100 bus was introduced in 1975. A typical S-100 system includes a single CPU, its program memory, and peripheral devices interfaced to the bus with one or more I/O boards. The CPU is directly involved in virtually every data exchange and processing task in such systems. Consequently, as system activity increases, the single system CPU is burdened more and more. System response time and throughput suffer as a result. Even the fastest, most efficient CPU must eventually yield when it is required to manage too many peripherals, or process too much data, or both.

There are two dominant factors which limit S-100 system throughput: a single system CPU, and a fixed bandwidth bus. Any attempt to increase system throughput must simultaneously address both of these limitations. Architectures which have multiple processors may help. If these processors compete with one another for use of the same fixed bandwidth bus to exchange data with their peripherals (e.g., a Multibus-like architecture), then system throughput is only incrementally improved. Ideally, each structured system task which would benefit from intelligent preprocessing and data buffering should have its own dedicated processor **and** its own dedicated bus. Then, each task processor would manage data transfers to and from its own peripheral(s), perform data buffering, formatting and preprocessing over its own independent expansion bus without competing for use of the main, or host, S-100 bus. As a result, the main, or host, processor would then be substantially offloaded, and free to manage the exchange of preprocessed data between itself and its intelligent system tasks in a structured and efficient manner over the host S-100 bus.

Cromemco's C-Bus architecture is designed to operate in precisely this ideal way; it exploits the inherent advantages of distributed processing, and it simultaneously overcomes the limitations of a fixed bandwidth bus in an S-100 setting. In short, Cromemco's C-Bus architecture brings **functional multiprocessing** to S-100 bus systems. The basic C-Bus system structure is illustrated in Figure 1.

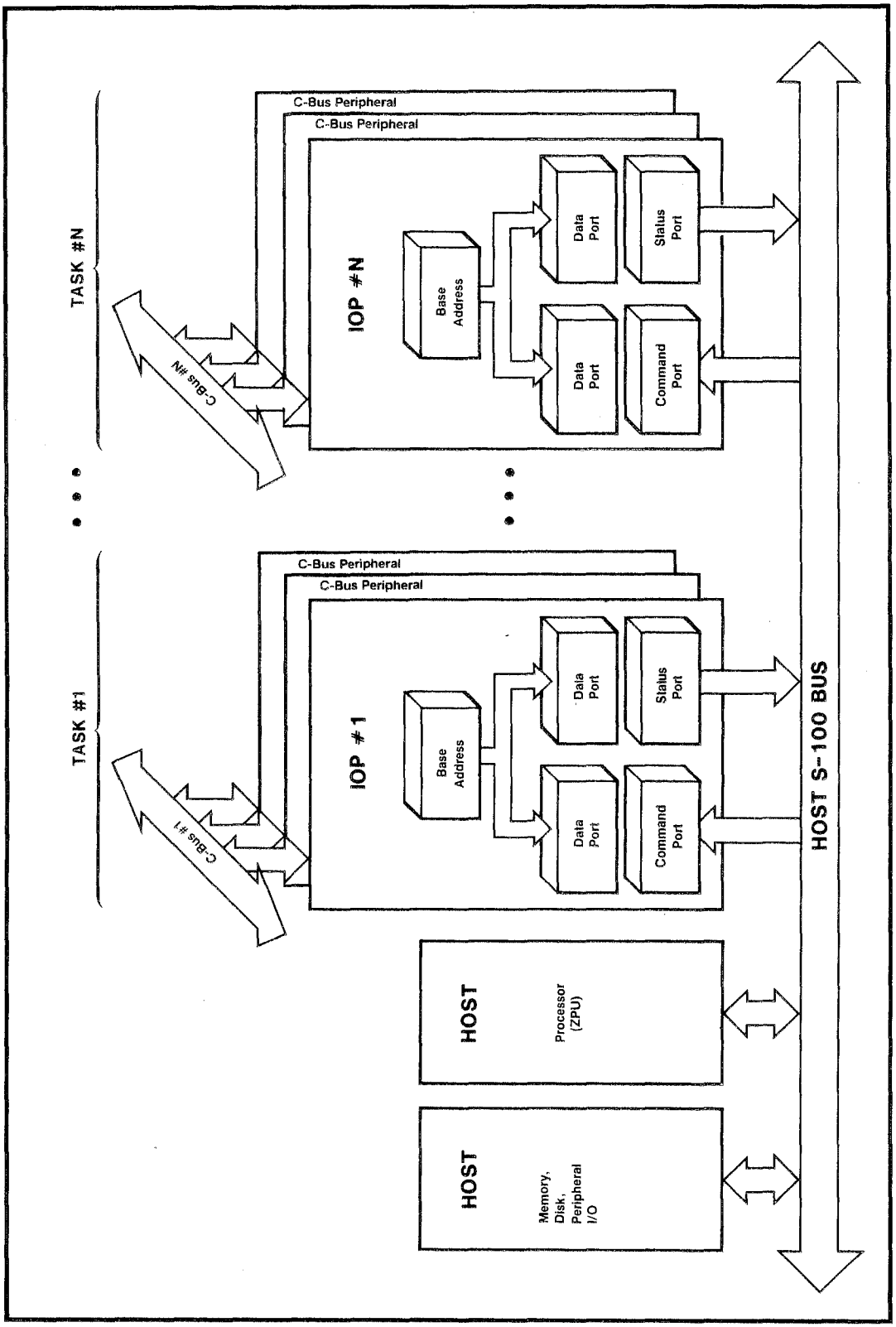


Figure 1: CROMEMCO'S C-BUS ARCHITECTURE

The figure shows a core S-100 system, consisting of a host S-100 bus, processor, memory and I/O boards, augmented by one or more **tasks.** Each task consists of an Input/Output Processor (IOP) board, an independent 50 conductor overhead expansion bus (a C-Bus), and the C-Bus peripherals connected to it (note that some tasks require only a single IOP board). The IOPs and C-Bus peripherals are standard dimensioned S-100 boards which occupy slots in the host S-100 bus, and draw power from it.

Each IOP is a single card computer which features a 4 MHz Z-80A CPU, 32 Kbytes of resident firmware (ROM) space, 16 Kbytes of RAM memory, and hardware interfaces to both the S-100 and C-Busses. The S-100 bus interface makes each IOP task look like two parallel I/O ports to the host processor; these ports may be mapped anywhere in the host processor's I/O space. The host passes commands and data to the IOP through the two ports in one direction, and the IOP passes status and preprocessed data back to the host in the other. A significant IOP feature not shown in the figure is the IOP's ability to interrupt the host and pass interrupt vectors to the S-100 bus.

The IOP C-Bus interface consists of a complete group of Z-80A address, data and control lines which gives the IOP and its peripherals full memory access, parallel I/O, interrupt and DMA capabilities without competing in any way with corresponding host S-100 bus functions. This gives the IOP programmer flexibility in designing a task structure for the most efficient gathering and processing of raw data by the IOP and C-Bus peripherals, and then passing the preprocessed data back to the host over the S-100 bus in either a polled or interrupt driven fashion.

The inherent flexibility of Cromemco's C-Bus architecture allows the system designer to create several useful variations on the S-100 system structure just described. For example, one or more IOP tasks and a host system may share the same physical volume, draw power from the same S-100 bus, but perform completely independent functions. Typical functions might include line multiplexing, data concentration, protocol conversion or data conversion performed by an IOP and one or more Quadart boards connected to it. Further variations include several independent but coresident IOP tasks running without host processor management, or even a stand alone IOP board acting as an intelligent I/O controller (the I/O devices could be controlled either over the IOP S-100 edge connector, or over the IOP's C-Bus interface).

Cromemco supports the growing C-Bus hardware line with a comprehensive set of software/firmware tools for IOP program development. These tools are supplied in Cromemco's **IOP Development Software** package, Model IDS, which includes:

1. **IOPMON**, a 2516 EPROM resident IOP monitor program which occupies IOP socket ROM0.

2. **IOPEX**, an IOP file management executive program which uploads/downloads host disk files to and from IOP memory.

3. **IOPDEBUG**, the IOP memory resident debugger program which expands the number of **IOPMON** commands.

4. Comprehensive documentation on the programs above, plus several other useful utility routines with documentation.

Chapter 2

IOP OPERATING INSTRUCTIONS

The IOP is a microprocessor controlled S-100 subsystem which must be viewed and understood in a system context. This chapter first defines that context, and then discusses the major IOP subsystems (except the IOP interrupt and DMA subsystems which are discussed in the following chapter) from an operational point of view. Most of the IOP configuration options are software defined -- see the **Control Register** and **Interrupt Register** bit descriptions in Appendix A near the end of this manual for a discussion of these options. The IOP features the following three hardware configuration options:

1. Selecting the IOP base address which places the IOP in the host's I/O map -- see Section 2.6,

2. Defining the IOP internal memory map with the contents of a memory mapping PROM -- see Section 2.5, and

3. Selecting the number of Wait States which are automatically inserted during IOP ROM read cycles -- see Section 2.4.

The IOP is factory shipped with: (1) IOP base address switch SW1 set to CFh, the switch setting shown in Figure 12, (2) a memory mapping PROM installed in board socket IC10 which defines the **standard IOP memory map** shown in Figure 6, and (3) with no Wait State jumper strap installed; this means that all IOP ROMs must have memory access times less than 570 nSec. These factory selected options anticipate the "standard IOP configuration" in which; (1) 2516 EPROM program **IOPMON** occupies IOP socket ROM0, (2) any additional firmware is programmed into 2516 EPROM devices which occupy IOP sockets ROM1 through ROM3, and (3) IOP ROM memory spans addresses 0000h - 1FFFh (8 Kbytes), and IOP RAM memory spans 4000h - 7FFFh (16 Kbytes). Carefully note that all Cromemco supplied programs in the **IOP Development Software** package assume that program **IOPMON** is installed in socket ROM0, and that IOP RAM memory spans 4000h - 7FFFh.

The reader is encouraged to exercise a newly purchased IOP board by following the check out procedure presented in Section 2.11, and to study Chapter 2 carefully before changing any of the factory selected configuration options.

2.1 IOP BLOCK DIAGRAM

The IOP is a single board computer with a Z-80A CPU clocked at 4.000 MHz, 16 Kbytes of 4116 dynamic RAM memory, socket space for up to 32 Kbytes of ROM memory, seven parallel I/O registers, and hardware interface circuitry to both the S-100 and C-Busses (see Figure 2).

From the viewpoint of the host S-100 bus, the IOP looks like two intelligent parallel I/O ports and a vectored interrupt request source. The two consecutively numbered ports may be switch mapped anywhere in the S-100 bus I/O space. From the viewpoint of peripherals connected to the C-Bus cable, the IOP looks like an intelligent controller which manages the C-Bus address, data and status lines, and which responds to vectored interrupt and DMA requests from the peripherals.

The IOP Z-80A executes program code from its own internal ROM or RAM memory (it can also execute program code from memory which is properly interfaced to its C-Bus cable); this program code in turn coordinates the exchange of data with C-Bus peripherals and the host processor and preprocesses the data traveling in both directions. In short, the program code operationally defines the **task** which the IOP performs. Applying unregulated +8 VDC to the IOP resets its Z-80A which causes it to automatically begin program execution at address 0000h. This is the starting address of ROM0, assuming the standard memory map. When IOP Monitor program **IOPMON** occupies this socket, a Power On Clear (POC) sequence automatically starts program **IOPMON** running. When the IOP is operated in a Cromix or CDOS host system, the host processor would then typically establish contact with **IOPMON**, download an IOP application program into IOP RAM memory, and start the program running.

The IOP uses seven internal parallel hardware registers to pass commands, status, interrupt vectors and data to and from the host over the S-100 bus, and also to control certain C-Bus events. The seven registers have fixed port addresses in the IOP's internal I/O map (OUT 00h - OUT 03h, and IN 00h - IN 02h). Four of the seven registers are also accessible to the host processor.

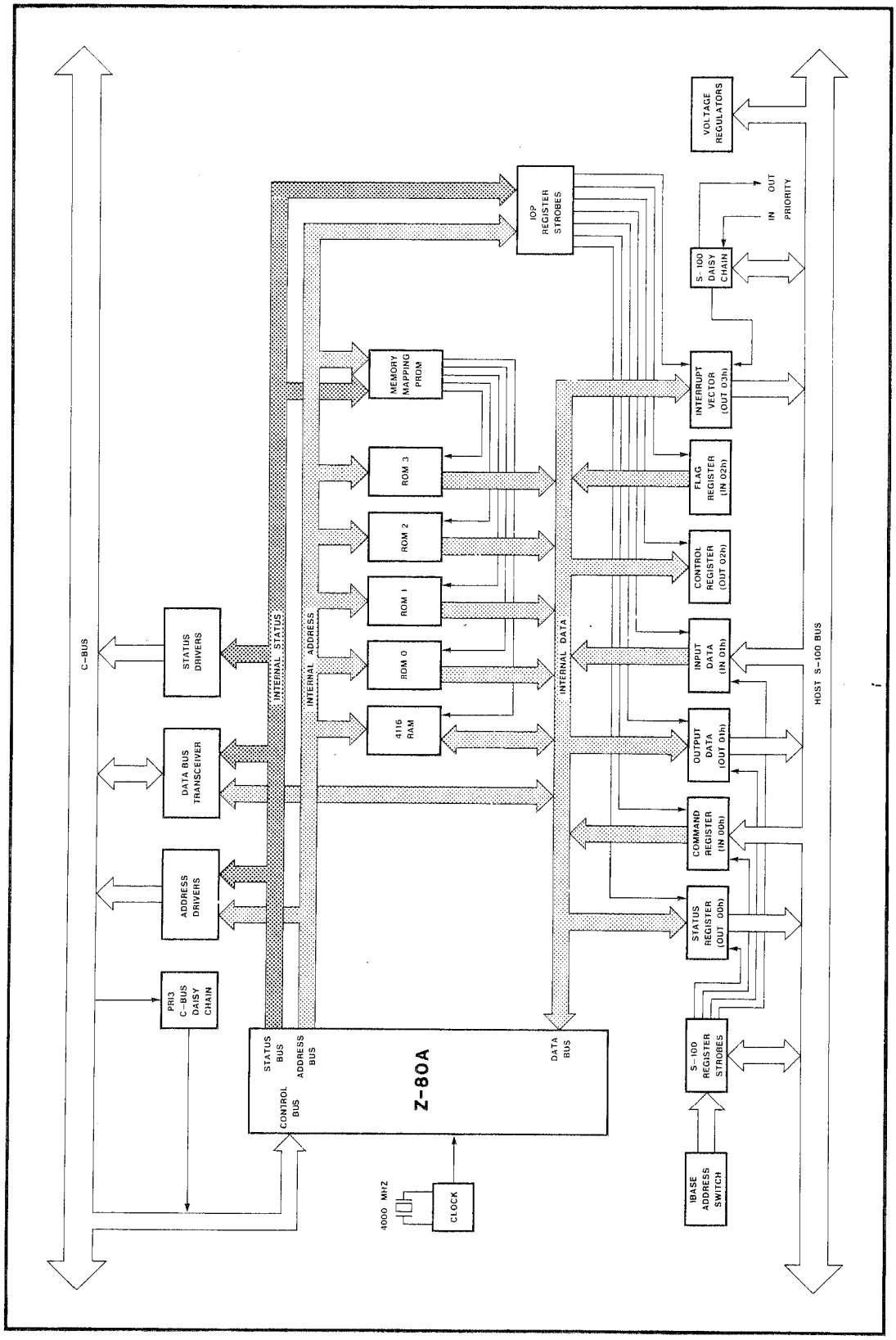


Figure 2: IOP BLOCK DIAGRAM

These four registers are positioned in the host I/O map by IOP base address (**Ibase**) switch SW1; the resulting port addresses then become OUT Ibase+00h, OUT Ibase+01h, IN Ibase+00h and IN Ibase+01h. Note that IN 00h corresponds to OUT Ibase+00h (**Command Register**), IN 01h corresponds to OUT Ibase+01h (**Input Data**), OUT 00h corresponds to IN Ibase+00h (**Status Register**), and OUT 01h corresponds to IN Ibase+01h (**Output Data**); that is, data output by the host is input by the IOP, and vice versa. A quick reference listing of the seven registers appears on the manual overleaf. Refer to Section 2.6 and Appendix A for a full discussion of these registers.

2.2 IOP CENTRAL PROCESSING UNIT

The heart of the IOP is its Z-80A Central Processing Unit (CPU). The Z-80A is an 8-bit CPU which can directly address 64 Kbytes of memory (addresses 0000h through FFFFh), 256 input ports and 256 output ports (port addresses 00h through FFh). The Z-80A's 158-member instruction set includes, as a subset, the 8080's 78-member instruction set. The Z-80A is clocked at a fixed, crystal controlled 4.000 MHz, yielding instruction execution times which range between 1.00 uSec (for 4-cycle instructions) to 5.75 uSec (for 23-cycle instructions) assuming that no Wait States are inserted. With the exceptions of input $\overline{\text{BUSRQ}}$ and output $\overline{\text{BUSAK}}$, the complete set of Z-80A address, data and control signals are brought out at the IOP C-Bus interface. Note that Z-80A status and control signals are similar to, but not identical with S-100 status and control signals, and that C-Bus peripherals make DMA requests using C-Bus line $\overline{\text{CPU DISCONNECT}}$, while $\overline{\text{BUS AVAILABLE}}$ serves as an acknowledgment.

The IOP Z-80A may be one of several operating system processors. Consequently, it is important to clearly delineate the boundaries between memories, I/O ports, interrupt and DMA request sources which are associated with each processor. Table 1 shows that the IOP Z-80A manages the C-Bus and the peripherals attached to it, as an isolated system, in much the same way as the host processor manages the S-100 bus.

Table 1

IOP/HOST CPU BOUNDARIES

EVENT	**IOP Z-80A**	**HOST CPU**
Memory Reads:	From Internal IOP Memory Or External C-Bus Memory	From S-100 Bus Memory
Memory Writes:	To Internal IOP Memory And External C-Bus Memory	To S-100 Bus Memory
I/O Reads:	From Internal IOP Registers Or External C-Bus Ports	From S-100 Bus Ports Including Two IOP Ports
I/O Writes:	To Internal IOP Registers And External C-Bus Ports	To S-100 Bus Ports Including Two IOP Ports
Interrupt Requests:	From Internal IOP Events Or C-Bus Peripherals	From S-100 Bus Peripherals Including the IOP
DMA Requests:	From C-Bus Peripherals	From S-100 Bus Peripherals Not Including the IOP

When the IOP Z-80A performs a memory read, then either internal IOP memory (as defined by the memory mapping PROM), or external C-Bus memory, is accessed. If internal and external memory addresses overlap, then IOP circuitry detects the conflict and routes only internal IOP memory data to the Z-80A. Memory writes, on the other hand, are performed to internal and external overlapping memory in parallel. Likewise, when the IOP Z-80A performs an I/O read, then either internal registers IN 00h through IN 02h, or external C-Bus ports are accessed. Again, overlapping port address reads are resolved in favor of internal IOP registers. IOP Z-80A I/O writes are directed to internal IOP registers OUT 00h through OUT 03h, and external C-Bus ports in parallel. The IOP Z-80A maskable $\overline{\text{INT}}$ pin may be driven active low either by internal IOP events (see Section 3.1), or by any C-Bus peripheral. Any C-Bus peripheral may make a software polled DMA request to the IOP Z-80A by driving C-Bus line $\overline{\text{CPU DISCONNECT}}$ active low (see Section 3.4).

2.3 IOP RAM MEMORY

The IOP is shipped with 16 Kbytes of read/write spanning addresses 4000h - 7FFFh. Refer to Section 2.5 for instructions on how to locate the RAM memory block elsewhere in the IOP memory map. IOP RAM memory physically consists of eight 4116 (16K x 1) dynamic RAM memory chips occupying sockets IC44 through IC47, and IC59 through IC62. IOP application programs with critical speed requirements should be loaded in, and run from IOP RAM memory since no CPU Wait States are inserted for these 150 nSec (max) memory access parts.

The 4116 dynamic memory chip must be **refreshed** periodically to maintain its data integrity. The **transparent refresh** technique is used on the IOP; that is, the seven bit refresh address supplied by the Z-80A during T2 and T3 of each M1 fetch cycle (while the Z-80A address bus is not being used for other purposes) is multiplexed onto all 4116 address lines and strobe $\overline{RAS}$ is pulsed low. Each such RAS only refresh cycle causes 128 bits of each 4116 device to be read and written back to their memory bit cells. An entire 4116 device is completely refreshed when the Z-80A has supplied 128 sequential RAS only refresh cycles. The 4116 must be completely refreshed every 2 mSec, or its data integrity is not guaranteed. The only way this requirement might not be met on the IOP is when the C-Bus $\overline{WAIT}$ line, <5>, is held active low for extended periods. This active signal holds off further Z-80A M1 cycles, and also refresh addresses as a consequence. Refer to Section 4.4 for a detailed discussion of this topic.

2.4 IOP ROM MEMORY

The IOP provides sockets ROM0 through ROM3 which may be occupied by 24-pin ROM devices conforming to the pin out shown in Figure 3. A unique **memory mapping PROM** allows a mix of device types to be plugged into the four sockets by supplying the high order address and control signals required by each device type. In addition, the memory mapping PROM locates the ROM sockets and the 16 Kbyte block of RAM memory in the internal IOP memory map. The factory shipped memory mapping PROM configures sockets ROM0 through ROM3 for TI 2516 (or Intel 2716) type devices, and it locates the four devices at 0000h-1FFFh in the IOP memory map (see Figure 6). This arrangement anticipates that 2516 program **IOPMON** will occupy IOP socket ROM0 in most applications.

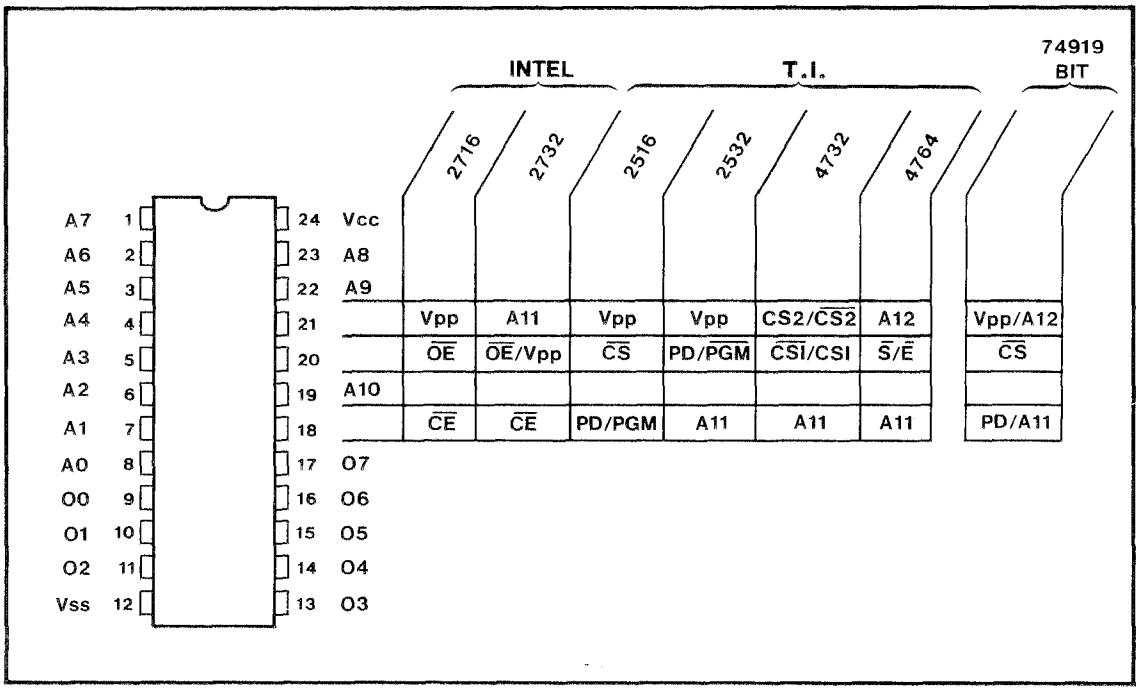


Figure 3: IOP ROM PIN OUTS

The IOP user may wish to program a custom memory mapping PROM to accommodate other ROM device types, and other map structures, in dedicated IOP applications. Section 2.5 describes how to determine the required memory mapping PROM contents for all of the following ROM devices, or their generic equivalents: Texas Instruments 2516 and 2532 EPROMs; Intel 2716 and 2732 EPROMs; Texas Instruments 4732 and 4764 masked ROMs.

All, none, or any combination of IOP ROM sockets may be occupied. Performing a memory read from an empty ROM socket almost always returns 0FFh data (all logic 1 bits from floating data bus lines). Writing to an empty IOP ROM socket has no affect other than consuming time, but writing to an occupied ROM socket **should be avoided** to preclude internal data bus conflicts between Z-80A write data and ROM read data.

The IOP provides three ROM memory speed options. The three options provide for: (1) inserting one Z-80A Wait State on every IOP ROM read cycle -- the factory shipped option, or (2) inserting one Z-80A Wait State on each M1

(memory fetch) cycle to IOP ROM memory, and inserting no Wait States on non-M1 cycles, or (3) inserting no Z-80A Wait States when reading IOP ROM memory. Note that regardless of the option selected, no Wait States are inserted during any IOP **RAM** read/write cycle, nor during any read/write cycle to external C-Bus memory. The selected option applies to **all** IOP ROM memory. Thus, when fast and slow ROM memory are mixed on the IOP, the option must meet the speed requirements of the slowest memory used. Table 2 below shows the required ROM maximum memory access times (address to valid data) when using each option. The **No Wait States** option is described only to anticipate future ROM speed improvements.

Table 2: IOP ROM MEMORY SPEED OPTIONS

MAXIMUM MEMORY ACCESS TIME (nSec)

ONE WAIT STATE INSERTED:	NEVER	M1 CYCLES ONLY	ALL READ CYCLES
2 Kbyte ROMs:	320	430	570
4 Kbyte ROMs:	240	350	490
8 Kbyte ROMs:	240	350	490

The options are selected by properly wiring the two solder jumper pads labeled **M2** on the IOP silk screen legend (see Figure 4). The IOP is shipped with no jumper installed which selects one Wait State on all IOP ROM read cycles so that the common 450 nSec 2516/2716 type EPROM may be used. If one of the other two options is desired, then the large plate heat sink must be removed to gain access to the solder pads below; loosen the three heatsink securing screws on the board solder side, then slide the slotted-guide heatsink assembly off the board.

Install a jumper strap between the two pads to select the **one Wait State each M1 cycle** option. To select the **no Wait States** option, connect the left pad (the one shown connected to IC4 pin 1) to any convenient ground point (like IC4 pin 7). Replace the heatsink when done.

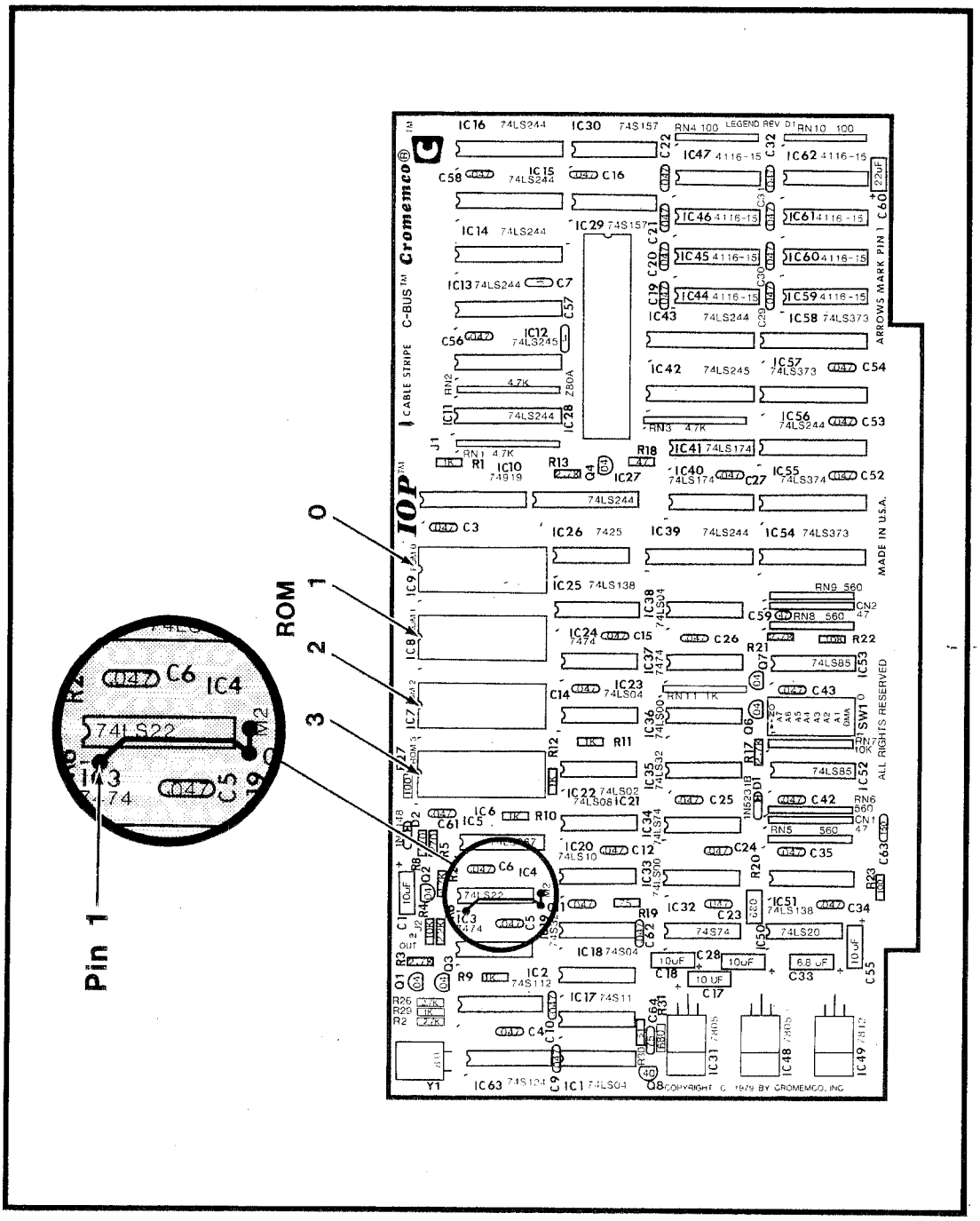


Figure 4: IOP ROM SPEED JUMPER PADS

2.5 IOP MEMORY MAPPING PROM

The IOP's memory mapping 74919 PROM (IC10 in the IOP Schematic Diagram) defines the IOP memory map. The factory shipped 74919 contents are shown in Figure 5 (all values shown in hexadecimal), and the resulting **standard IOP memory map** is shown further below in Figure 6. The following paragraphs describe how to program 74919 PROMs to accommodate different ROM device types, and to produce other IOP memory maps. The reader may skip this section entirely if the standard memory map is used.

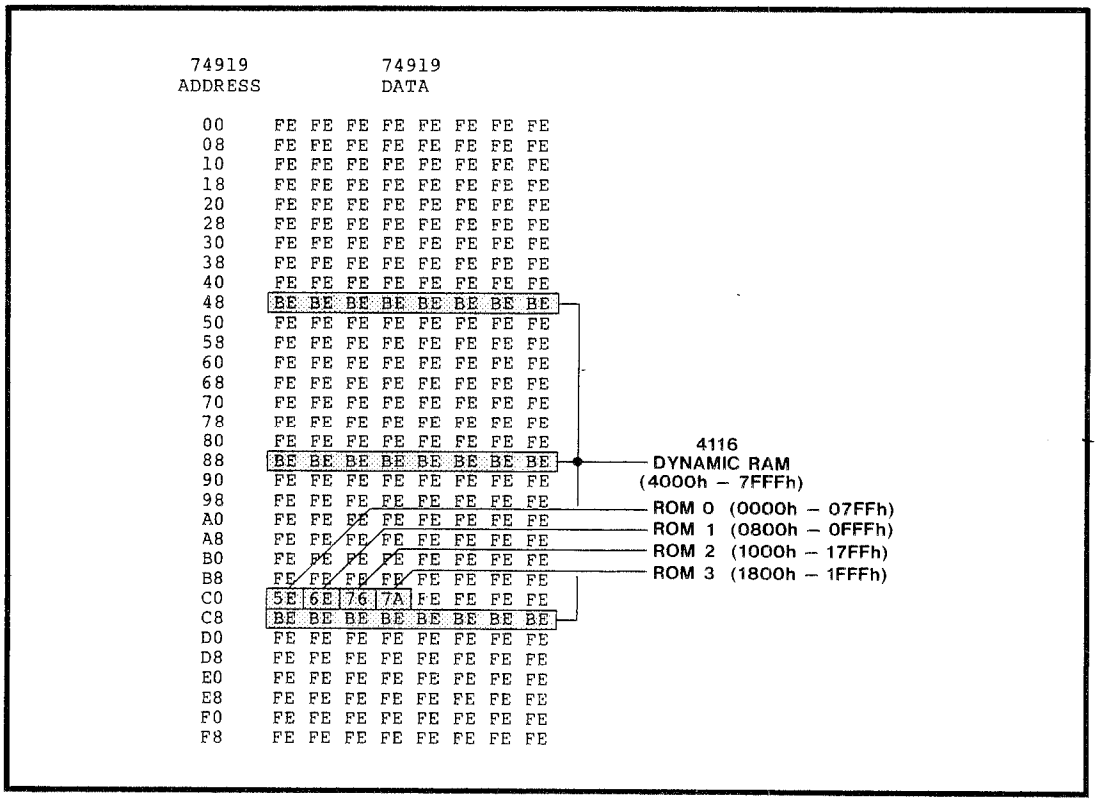


Figure 5: FACTORY SHIPPED 74919 CONTENTS

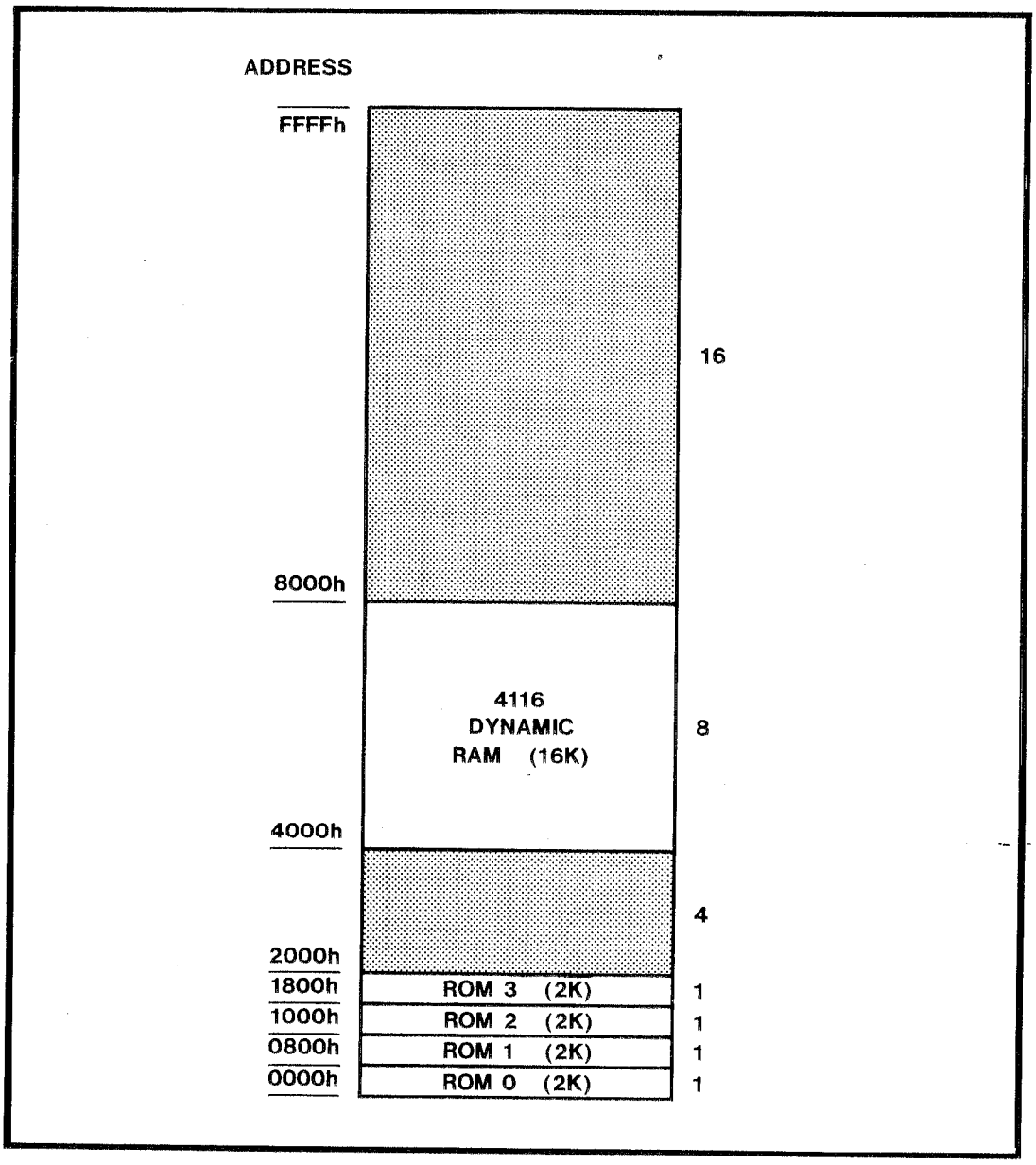


Figure 6: RESULTING IOP STANDARD MEMORY MAP

The basic function of the 74919 is to convert IOP address and status input data into RAM and ROM select output data by means of its programmed contents. Figure 7 shows that a unique 74919 byte is addressed for every combination of IOP control bus lines $\overline{WR}$, $\overline{RFSH}$, $\overline{MREQ}$ and address lines A15 - A11. Since only A15 through A11 are used as inputs, each programmed 74919 byte corresponds to a 2 Kbyte block of internal IOP memory (individual bytes are addressed by A10 - A0). If, for example, $\overline{WR}$ = 0, $\overline{RFSH}$ = 1, $\overline{MREQ}$ = 0, A15 = A14 = A13 = 0 and A12 = A11 = 1, then the IOP Z-80A is performing a memory ($\overline{MREQ}$ = 0) write cycle ($\overline{WR}$ = 0) to memory block 1800 - 1FFF.

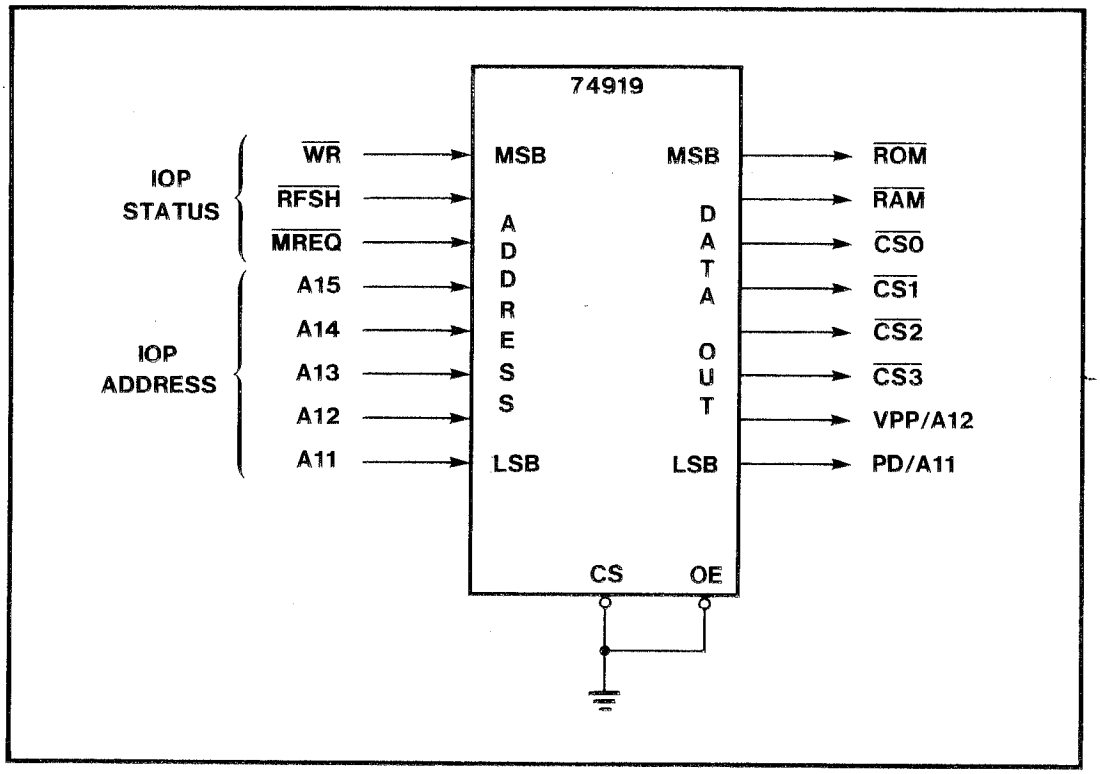


Figure 7: 74919 PROM INPUTS AND OUTPUTS

Status signals $\overline{WR}$, $\overline{RFSH}$ and $\overline{MREQ}$ drive the 74919 high order address lines, and consequently they partition the 74919 memory map into the eight regions shown to the right in Figure 8. Of the eight regions, only three correspond to actual IOP memory cycles: $\overline{WR}$ = 0, $\overline{RFSH}$ = 1 and $\overline{MREQ}$ = 0 for RAM write cycles; $\overline{WR}$ = 1, $\overline{RFSH}$ = 0 and $\overline{MREQ}$ = 0 for RAM refresh cycles; and $\overline{WR}$ = 1, $\overline{RFSH}$ = 1 and $\overline{MREQ}$ = 0 for RAM or ROM read cycles.

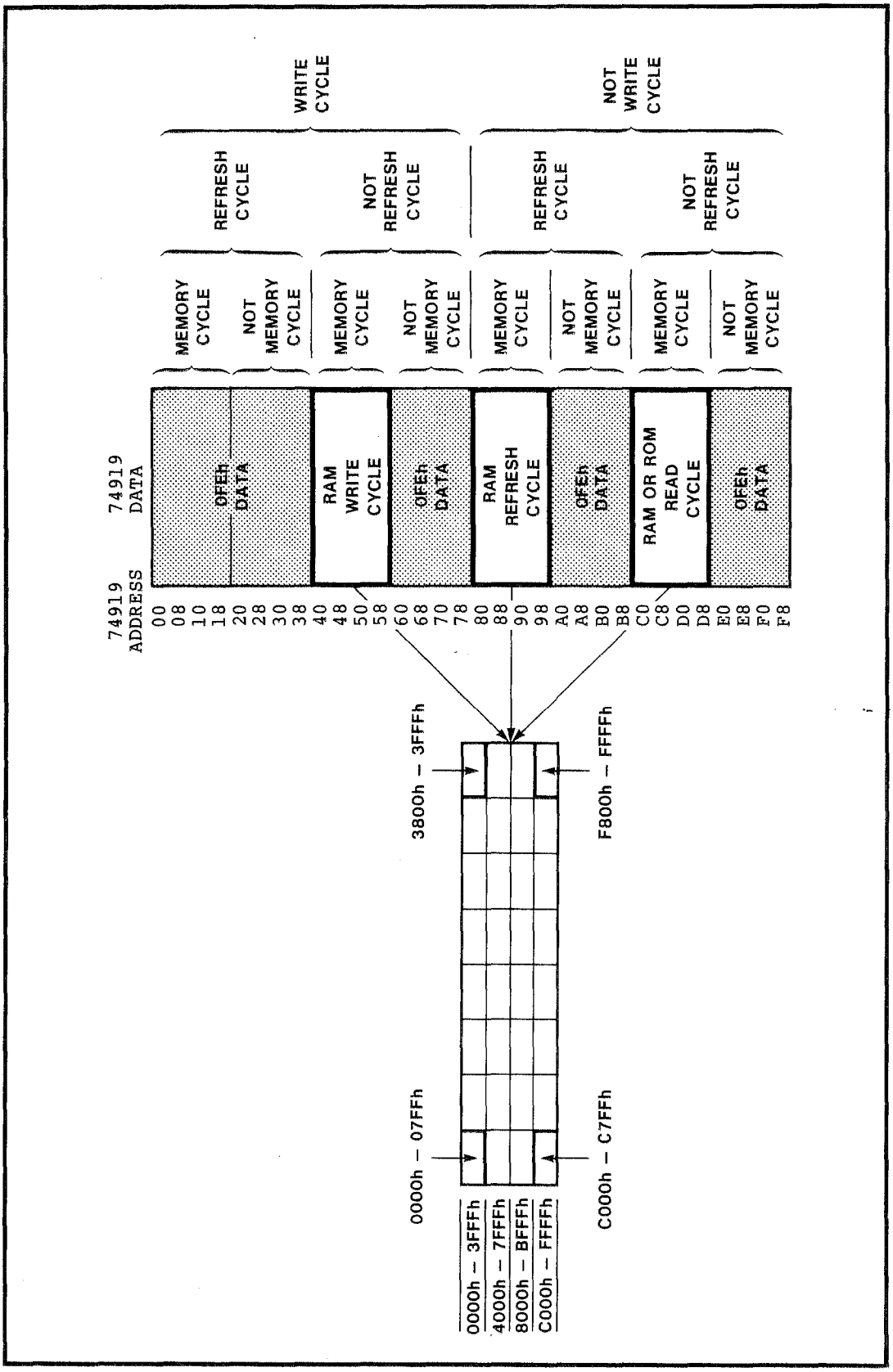


Figure 8: 74919 MAP PARTITIONS

IOP address lines A15 through A11 further partition the 74919 map as shown to the left in Figure 8. Each of the three regions mentioned above spans the entire 64 Kbyte memory address space with 32 programmable bytes, making each byte correspond to a different 2 Kbyte memory block. Reference to Figure 7 shows that the 74919 supplies eight memory select output data bits for each input combination. Bit $\overline{ROM}$ is programmed to logic 0 only if a ROM chip resides at the addressed 2 Kbyte block. In this case, one of the four ROM chip select bits, $\overline{CS0}$ through $\overline{CS3}$, would also be programmed to logic 0 to enable the chip's socket, ROM0 through ROM3. If, in addition, a 32 (or 64) Kbit ROM is used, then two (or four) consecutive 74919 bytes would be programmed with bit $\overline{ROM}$ = 0, $\overline{CS}$ = 0 (the same chip select line in all bytes), and with bit A11 (and A12) progressing through a binary count to access successive 2 Kbyte blocks of memory on the ROM chip. Note that ROM memory would typically be mapped into the lowest 2 Kbyte region of the IOP memory map anticipating automatic program start up after a POC.

Bit PD/A11 serves as a Power Down input on 2516 type devices, and as an A11 address line on 4 and 8 Kbyte ROMs. If any 2516 type EPROMs are used on the IOP, **then all 74919 PD/A11 bits must be programmed to logic 0.** This means that either **all** or **none** of the IOP ROMs must be 2516 type devices. Programming all PD/A11 bits to logic 0 permanently enables all 2516 type devices, and as a result, the 2516 address to data access time limits the ROM memory access speed. If, on the other hand, the PD/A11 line were to be strobed low as a 2516 is addressed, then the PD low to data access time would limit the ROM memory access speed far below the 2 Kbyte ROM values listed in Table 2 above.

Bit $\overline{RAM}$ is programmed to logic 0 only if internal IOP 4116 dynamic memory resides as the addressed 2 Kbyte memory block. In this case, eight sequential 74919 bytes should be programmed with $\overline{RAM}$ = 0; the same address positions in 74919 partitions **RAM write cycles, RAM refresh cycles** and **RAM read cycles** must also be programmed with bit $\overline{RAM}$ = 0. When bit $\overline{RAM}$ is programmed to logic 0, all other bit positions in the same byte should be programmed to logic 1 except when 2516 type devices are used; then PD/A11 should be programmed to logic 0.

Example

Assume that internal IOP memory is to consist of 4116 dynamic RAM memory from 0000h - 3FFFh and four 2516 EPROMs in sockets ROM0 through ROM3 spanning E000h - FFFFh. A 74919 bipolar PROM would then be programmed with the data shown in Figure 9, and inserted in IOP socket IC10 to yield the IOP memory map shown in the same figure.

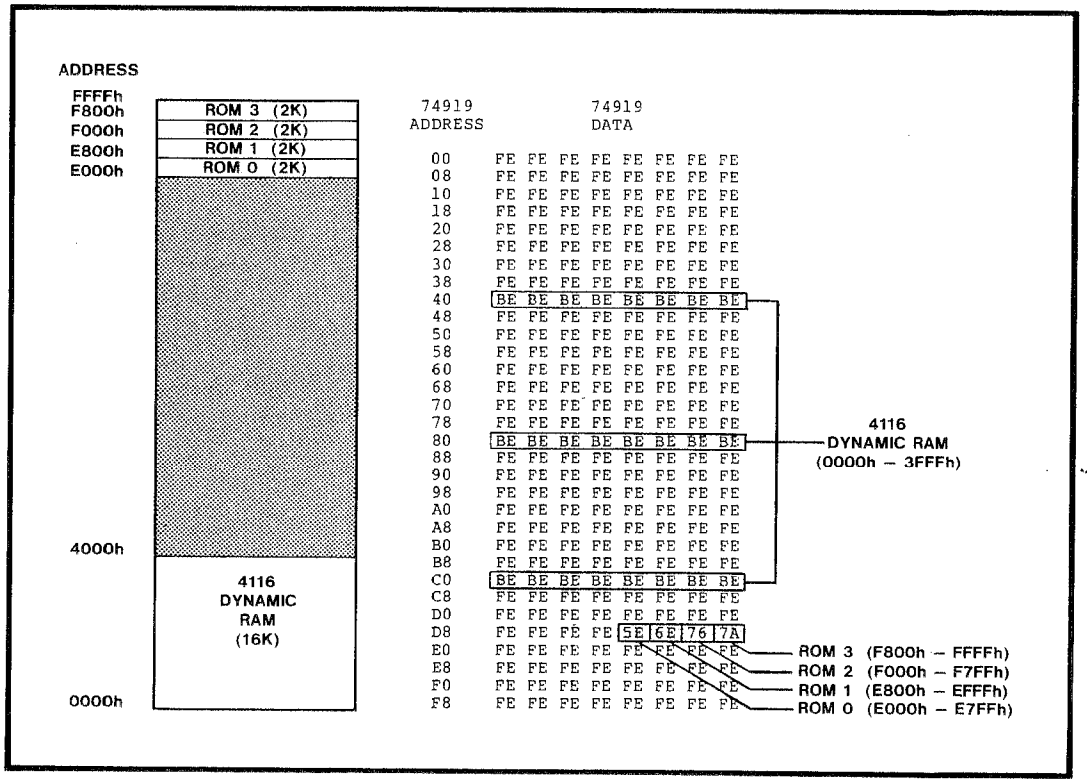


Figure 9: FIRST CUSTOM MAP AND 74919 DATA

Example

Assume that internal IOP memory is to consist of two 4764 masked ROMs spanning 0000h - 3FFFh in sockets ROM0 and ROM1, then 4116 dynamic RAM memory from 4000h - 7FFFh, then two 4732 masked ROMs spanning 8000h - 9FFFh in sockets ROM2 and ROM3. The 74919 PROM contents and resulting memory map are shown below in Figure 10.

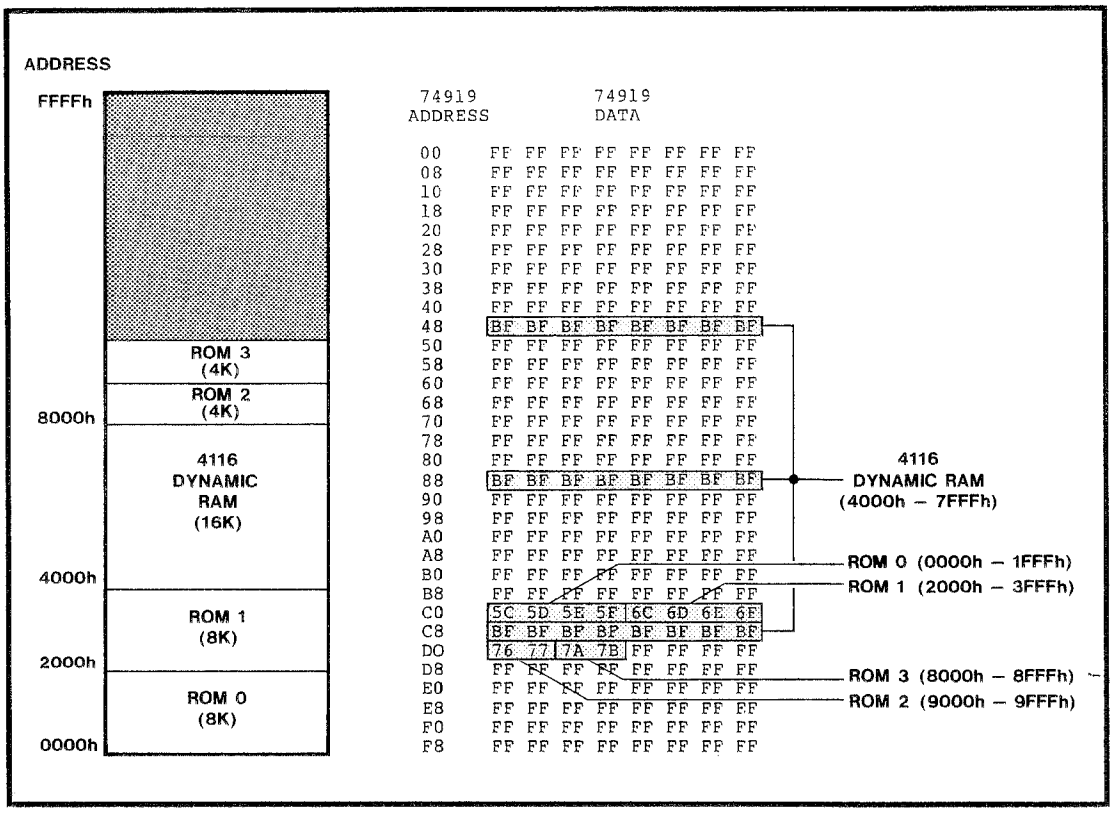


Figure 10: SECOND CUSTOM MAP AND 74919 DATA

2.6 IOP I/O REGISTERS

Seven internal IOP registers are provided to: (1) exchange data, status and commands between the IOP Z-80A and the host processor, (2) supply interrupt vectors in response to Interrupt Acknowledge cycles from the host processor, and (3) manage C-Bus DMA requests and acknowledgments. The IOP Z-80A has input/output access to all seven registers, while the host processor has direct input/output access to only four: **Status Register, Output Data, Command Register,** and **Input Data** (see manual overleaf for register summary). The port addresses of these seven registers are fixed in the internal IOP I/O map (IN 00h - IN 02h, and OUT 00h - OUT 03h), but are switch located in the S-100 bus I/O map. This arrangement is illustrated in Figure 11.

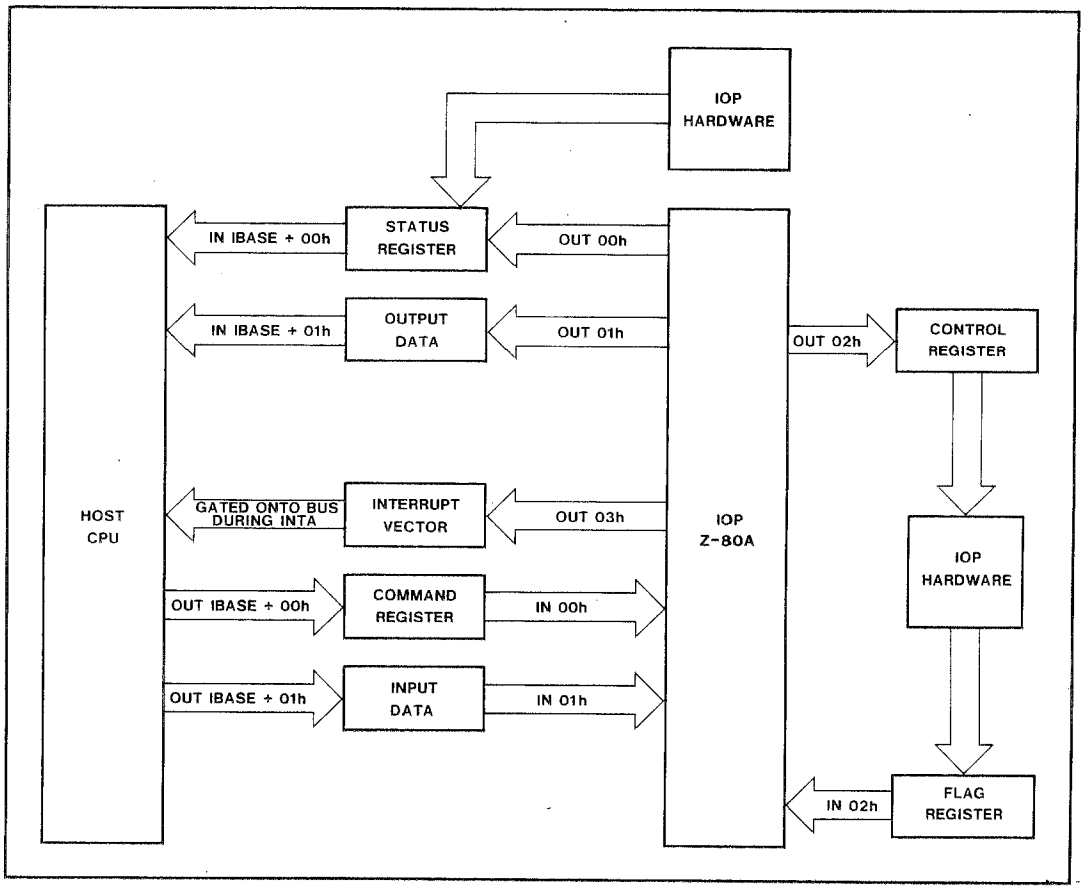


Figure 11: IOP REGISTER ACCESS

IOP switch SW1 defines the **IOP Base Address** (or **Ibase**) in the host I/O map. The seven address switches (A7 is the MSB and A1 is the LSB) generate 128 even Ibase values (IOP hardware assumes A0=0). Figure 12 shows SW1 set to make Ibase = CEh; the host processor would then communicate with the IOP using I/O ports CEh and CFh.

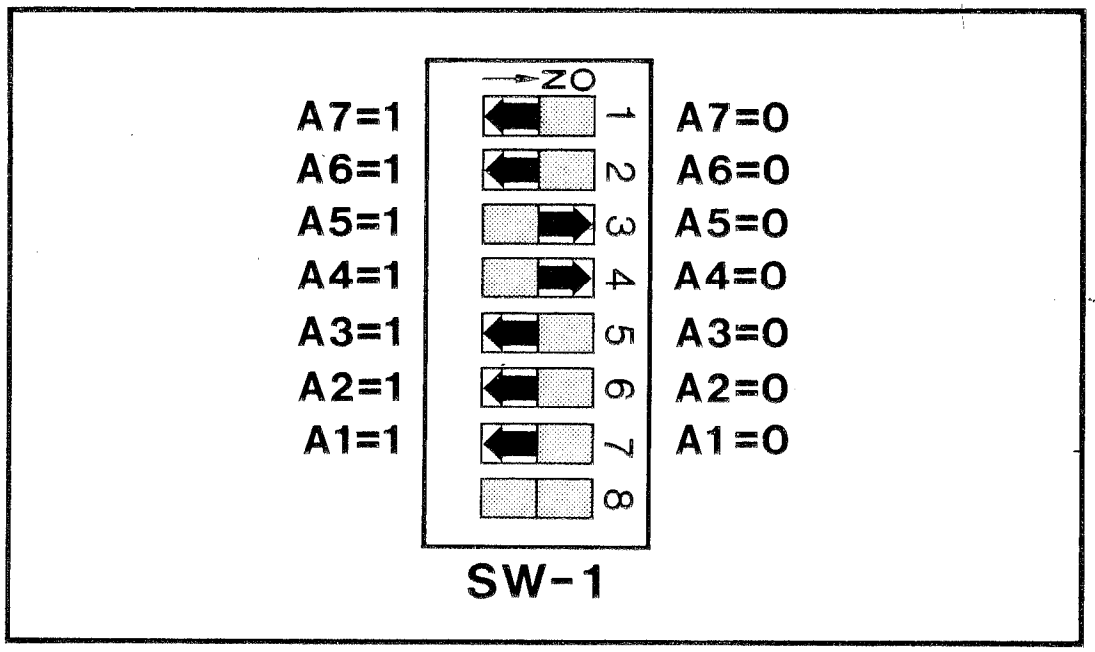


Figure 12: IOP BASE ADDRESS SWITCH, SW1

All Cromemco **host resident** IOP software assumes that Ibase is CEh for single IOP systems; BEh, AEh and 9Eh are assumed in sequence for multiple IOP systems. In addition to IOP port assignments, Figure 13 illustrates the port numbering convention assumed in all Cromemco supplied software. It is strongly recommended that the IOP user follow this port numbering.

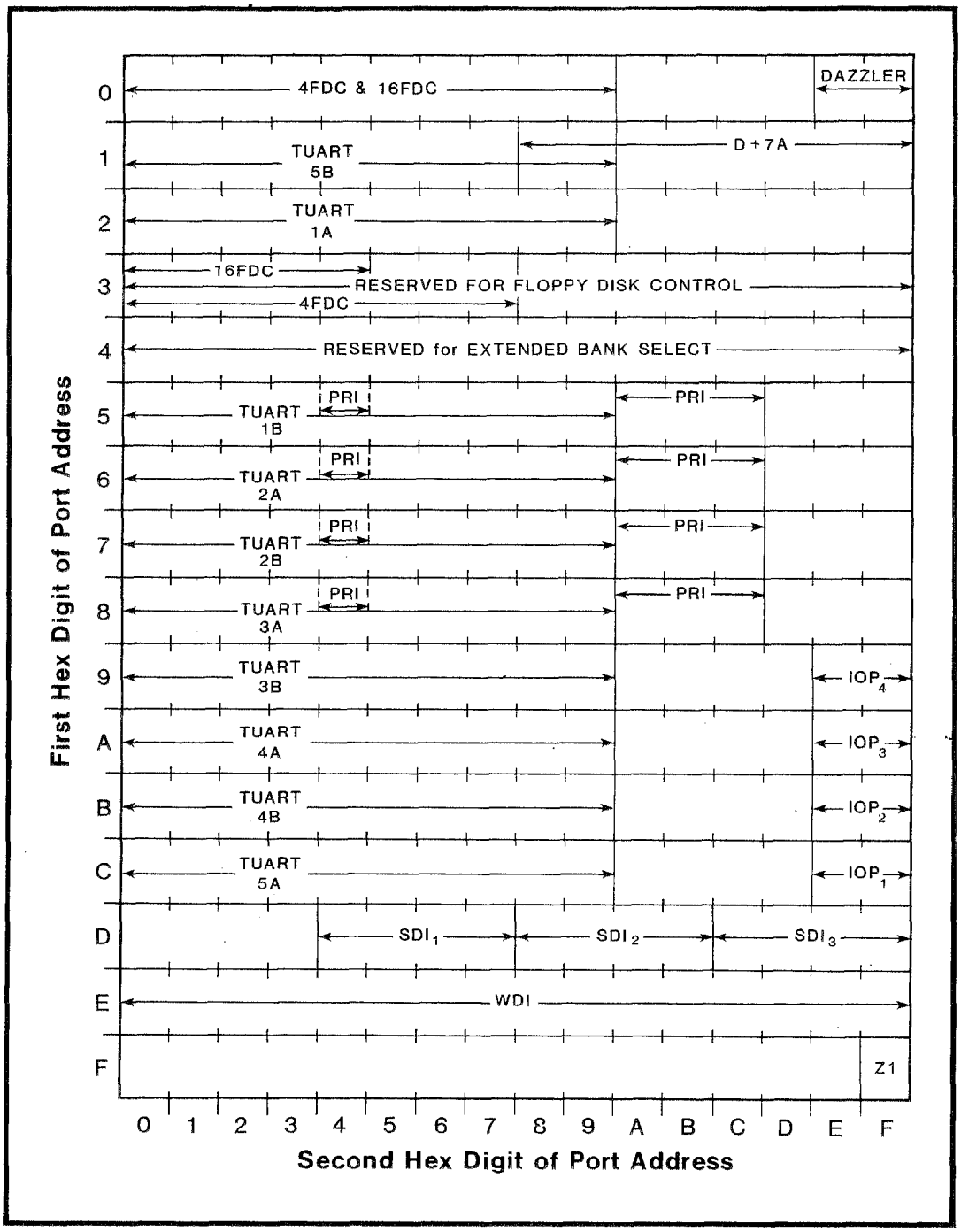


Figure 13: CROMEMCO PORT ASSIGNMENT CONVENTION

The labels assigned to individual IOP register bits are shown on the manual overleaf. Appendix A discusses each bit function in detail, so that material will not be repeated here. Section 2.9 illustrates through programming examples how to use the **Status Register, Output Data, Command Register, Input Data** and **Flag Register** to establish and manage a communication link between the host and IOP without using interrupts. Section 3.3 illustrates how the **Control Register** and **Interrupt Vector Register** are used in an interrupt driven host/IOP link. Section 3.4 discusses **Control Register** functions which relate to C-Bus DMA transfer.

2.7 IOP RESET AND POWER ON CLEAR STATE

Three sources may reset the IOP board (see Figure 14):

1. A Power On Clear (POC), or
2. Setting **Control Register** bit **S-100 Reset** (D3), or
3. Setting **Control Register** bit **C-Bus Clear** (D5).

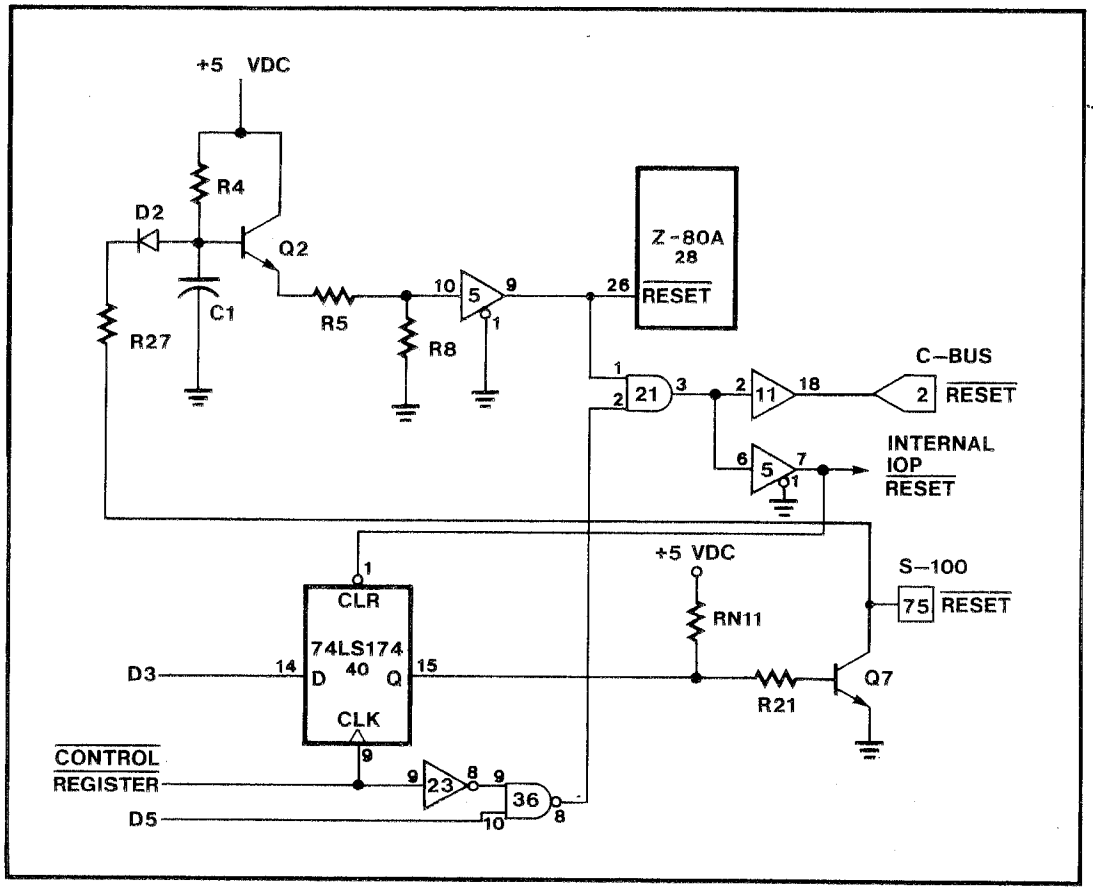


Figure 14: IOP POC AND RESET CIRCUITRY

Applying +8 VDC (unregulated) to the IOP board causes C1, R4, Q2 and associated circuitry to generate a momentary low level POC pulse at IC5 pin 9. This pulse resets the IOP Z-80A, resets the IOP internal circuitry (see Figure 15), and forces C-Bus line $\overline{\text{RESET}}$ active low.

Outputting a byte to the **Control Register** with bit **S-100 Reset** (D3) set forces S-100 bus line $\overline{\text{RESET}}$ active low and again forces IC5 pin 9 low momentarily. This level again resets the IOP's Z-80A, resets the IOP internal circuitry (including bit **S-100 Reset** at IC40 pin 1 -- see Figure 15), and forces C-Bus line $\overline{\text{RESET}}$ active low.

Outputting a byte to the **Control Register** with bit **C-Bus Clear** (D5) set generates a momentary low level at IC36 pin 8. This level resets the IOP internal circuitry (see Figure 15) and forces C-Bus line $\overline{\text{RESET}}$ active low. Unlike the previous two cases, the IOP's Z-80A is **not** reset, so IOP program execution continues uninterrupted.

Figure 15 illustrates the effect that any one of these three reset sources has on the IOP internal circuitry. Register bits which are not affected appear with ? entries (random after a POC and unaltered by a **S-100 Reset** and **C-Bus Clear**), and **n/a** entries mean **not applicable.**

The IOP Z-80A is reset by either a POC or by setting **Control Register** bit **S-100 Reset,** as mentioned above. The Z-80A responds to an active low level on its $\overline{\text{RESET}}$ pin by:

1. disabling maskable interrupts,
2. resetting its I-Register (IM2 Interrupt Address Register) to 00h,
3. resetting its R-Register (Refresh Address Register) to 00h,
4. activating IM0 (Interrupt Mode 0), and
5. commencing program execution at IOP memory location 0000h.

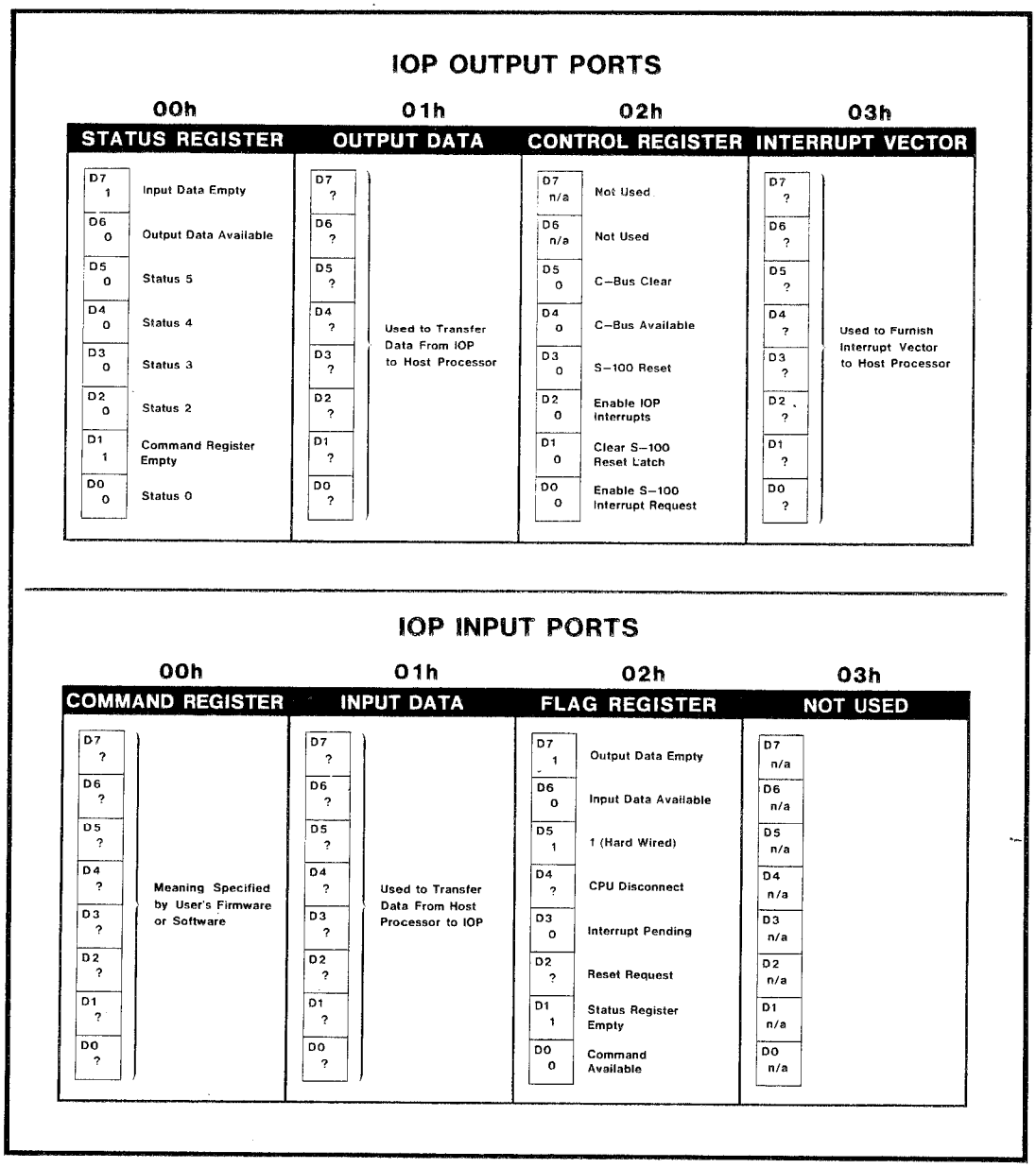


Figure 15: IOP REGISTERS AFTER POC, RESET OR CLEAR

A ROM resident initialization routine would typically be mapped to begin at IOP memory location 0000h in end product applications which require automatic start up after power is applied (see Sections 2.4 and 2.5).

The IOP can reset the host system by forcing S-100 bus line $\overline{\text{RESET}}$ low, but the host system **cannot** reset the IOP. In particular, the IOP **cannot** be reset by either S-100 bus line $\overline{\text{RESET}}$ nor by S-100 bus line $\overline{\text{POC}}$; the IOP generates its own POC strobe. The IOP may determine that an S-100 bus reset has occurred by polling **Flag Register** bit **Reset Request**, however, and optionally reset itself in response. This flexible arrangement assures that critical IOP controlled I/O operations are not interrupted by S-100 bus resets.

2.8 IOP CONNECTORS, PIN OUTS AND CABLES

There are three connectors on the IOP board: (1) an S-100 bus edge connector, (2) J1, a 50-conductor C-Bus connector, and (3) J2, a 2-conductor S-100 bus daisy chain connector.

Table 3 lists the S-100 bus signal lines which are used by the IOP, and Figure 16 shows the physical pin numbering convention. The subset of S-100 bus lines used is sufficient for passing I/O data between the host CPU and the IOP, for supplying interrupt requests and vectors to the host CPU, and for supplying power to the IOP. S-100 bus lines which are not connected to the IOP appear with dashed (---) entries in the table.

C-Bus cabling connects to the IOP board at J1 as shown in Figure 17. The C-Bus cable assembly physically consists of a 50-conductor ribbon cable and with either two plugs (an IOP and one C-Bus peripheral -- part number 519-0100), four plugs (an IOP and up to three C-Bus peripherals -- part number 519-0063), and five plugs (an IOP and up to four C-Bus peripherals -- part number 519-0101).

Table 4 lists C-Bus signal names with matching pin numbers. All references to **GND** in the table imply a direct connection to S-100 bus line GND, at 0 VDC potential, and again, dashed table entries mean no connection exists.

Table 3: IOP S-100 BUS CONNECTIONS

Pin	Signal	Pin	Signal	Pin	Signal	Pin	Signal
[1]	+8 VDC	[26]	pHLDA	[51]	+8 VDC	[76]	---
[2]	+18 VDC	[27]	---	[52]	-18 VDC	[77]	$\overline{\text{pWR}}$
[3]	---	[28]	---	[53]	---	[78]	pDBIN
[4]	---	[29]	A5	[54]	---	[79]	A0
[5]	---	[30]	A4	[55]	---	[80]	A1
[6]	---	[31]	A3	[56]	---	[81]	A2
[7]	---	[32]	---	[57]	---	[82]	A6
[8]	---	[33]	---	[58]	---	[83]	A7
[9]	---	[34]	---	[59]	---	[84]	---
[10]	---	[35]	DO1	[60]	---	[85]	---
[11]	---	[36]	DO0	[61]	---	[86]	---
[12]	---	[37]	---	[62]	---	[87]	---
[13]	---	[38]	DO4	[63]	---	[88]	DO2
[14]	---	[39]	DO5	[64]	---	[89]	DO3
[15]	---	[40]	DO6	[65]	---	[90]	DO7
[16]	---	[41]	DI2	[66]	---	[91]	DI4
[17]	---	[42]	DI3	[67]	---	[92]	DI5
[18]	---	[43]	DI7	[68]	---	[93]	DI6
[19]	---	[44]	sM1	[69]	---	[94]	DI1
[20]	---	[45]	sOUT	[70]	---	[95]	DI0
[21]	---	[46]	sINP	[71]	---	[96]	sINTA
[22]	---	[47]	---	[72]	---	[97]	---
[23]	---	[48]	---	[73]	$\overline{\text{INT}}$	[98]	---
[24]	---	[49]	---	[74]	---	[99]	---
[25]	---	[50]	GND	[75]	$\overline{\text{RESET}}$	[100]	GND

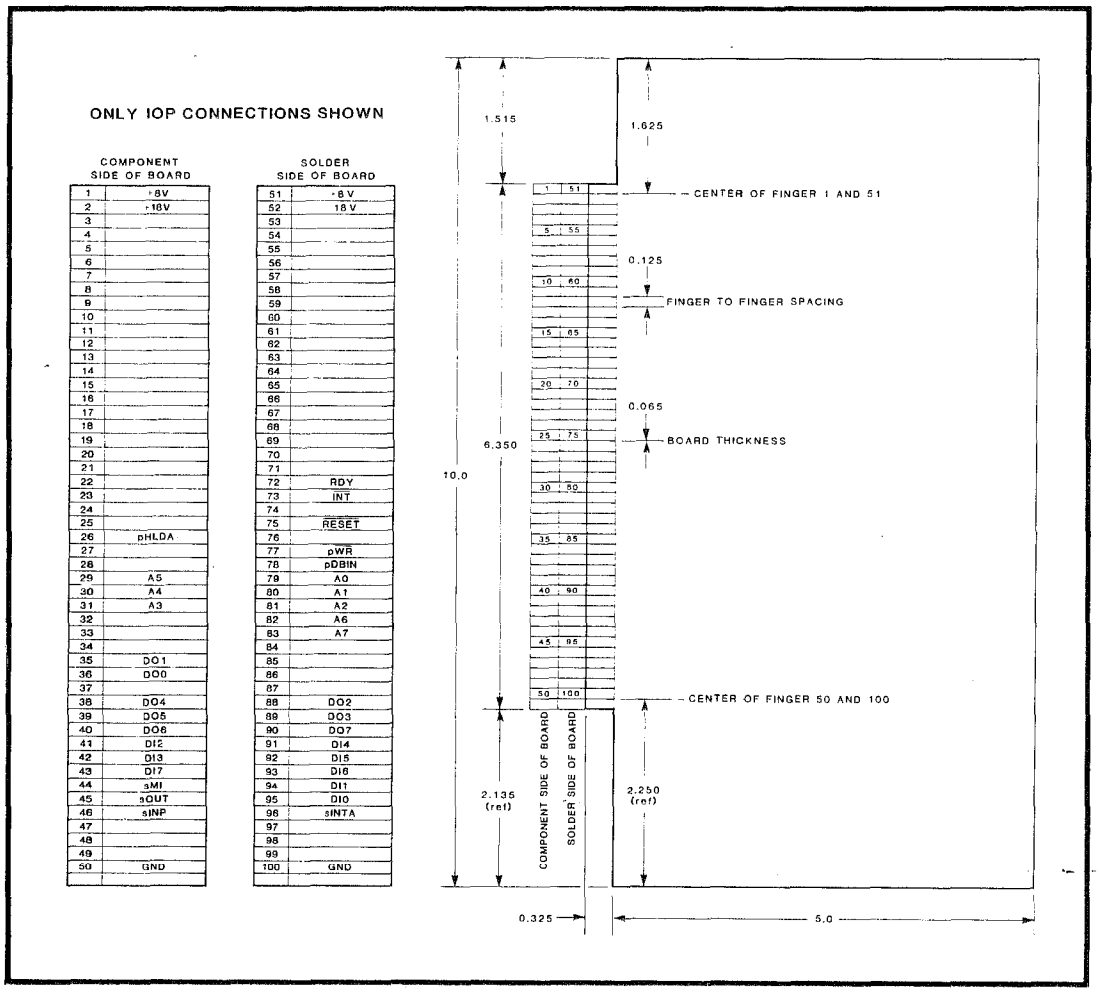


Figure 16: IOP S-100 BUS NUMBERING, DIMENSIONS

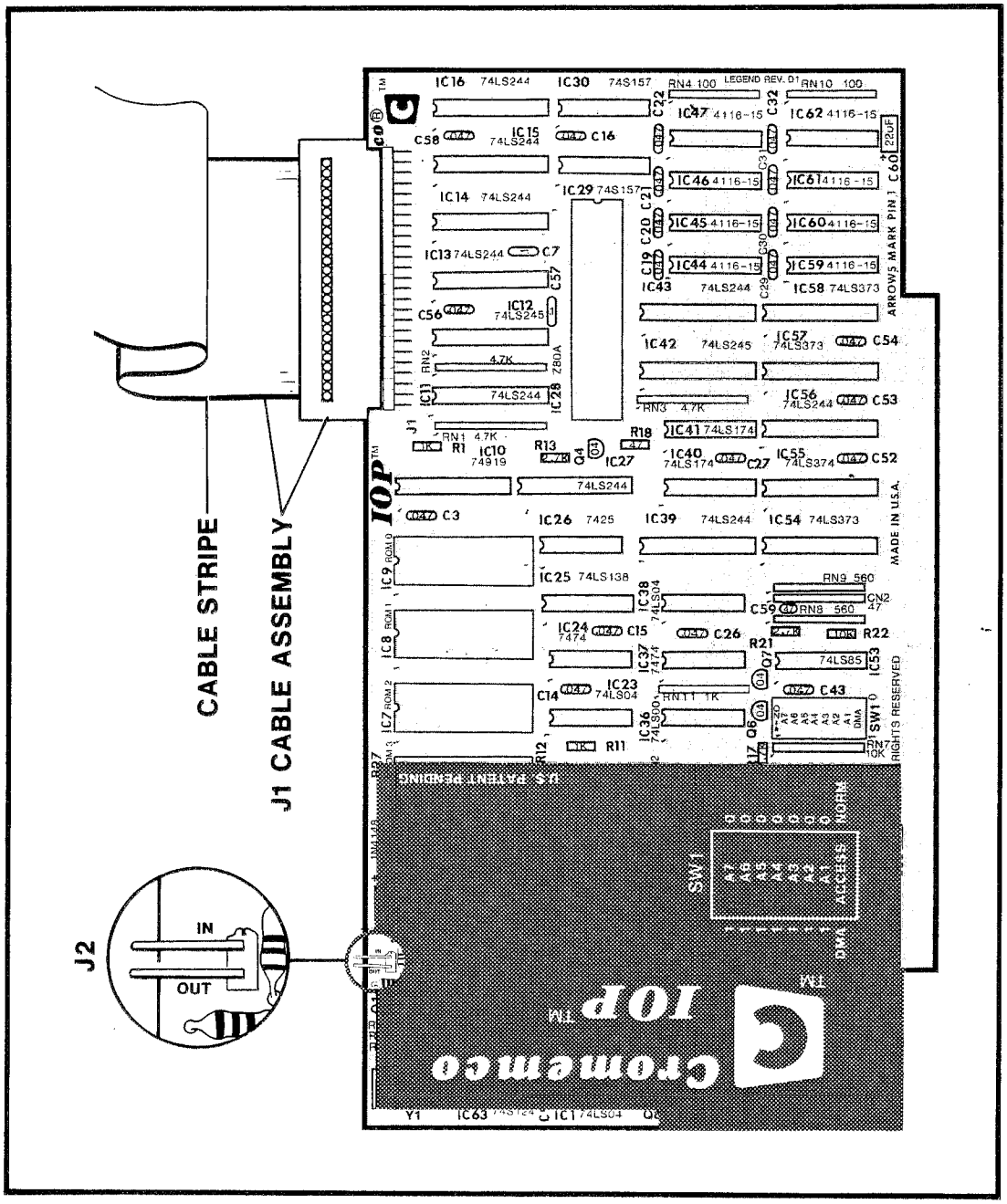


Figure 17: IOP CONNECTORS J1 AND J2

Table 4: IOP C-BUS CONNECTIONS

Pin	Signal	Pin	Signal
<1>	GND	<26>	A10
<2>	$\overline{\text{RESET}}$	<27>	A11
<3>	$\overline{\phi}$	<28>	GND
<4>	GND	<29>	A12
<5>	$\overline{\text{WAIT}}$	<30>	A13
<6>	D0	<31>	A14
<7>	D1	<32>	A15
<8>	D2	<33>	$\overline{\text{RD}}$
<9>	D3	<34>	GND
<10>	D4	<35>	$\overline{\text{WR}}$
<11>	D5	<36>	$\overline{\text{M1}}$
<12>	D6	<37>	$\overline{\text{MREQ}}$
<13>	D7	<38>	$\overline{\text{IORQ}}$
<14>	GND	<39>	$\overline{\text{RFSH}}$
<15>	A0	<40>	$\overline{\text{CPU DISCONNECT}}$
<16>	A1	<41>	$\overline{\text{BUS AVAILABLE}}$
<17>	A2	<42>	$\overline{\text{HALT}}$
<18>	A3	<43>	GND
<19>	A4	<44>	$\overline{\text{INT}}$
<20>	A5	<45>	$\overline{\text{NMI}}$
<21>	GND	<46>	---
<22>	A6	<47>	---
<23>	A7	<48>	$\overline{\text{PRI 3}}$
<24>	A8	<49>	---
<25>	A9	<50>	GND

Figure 18 shows the pin numbering convention used for IOP connector J1. Section 4.11 discusses C-Bus signals and waveforms in detail. Note that S-100 bus lines are called out in Table 3 with square brackets ([]) and C-Bus lines in Table 4 with angular brackets (<>). This convention will be used consistently throughout the manual to avoid confusion over duplicate bus line names.

Figure 17 above shows the location of IOP connector J2 which brings out two lines: S-100 bus daisy chain $\overline{\text{PRIORITY OUT}}$ and $\overline{\text{PRIORITY IN}}$. Section 3.2 discusses how to use this connector to define the IOP interrupt task priority among other S-100 bus tasks.

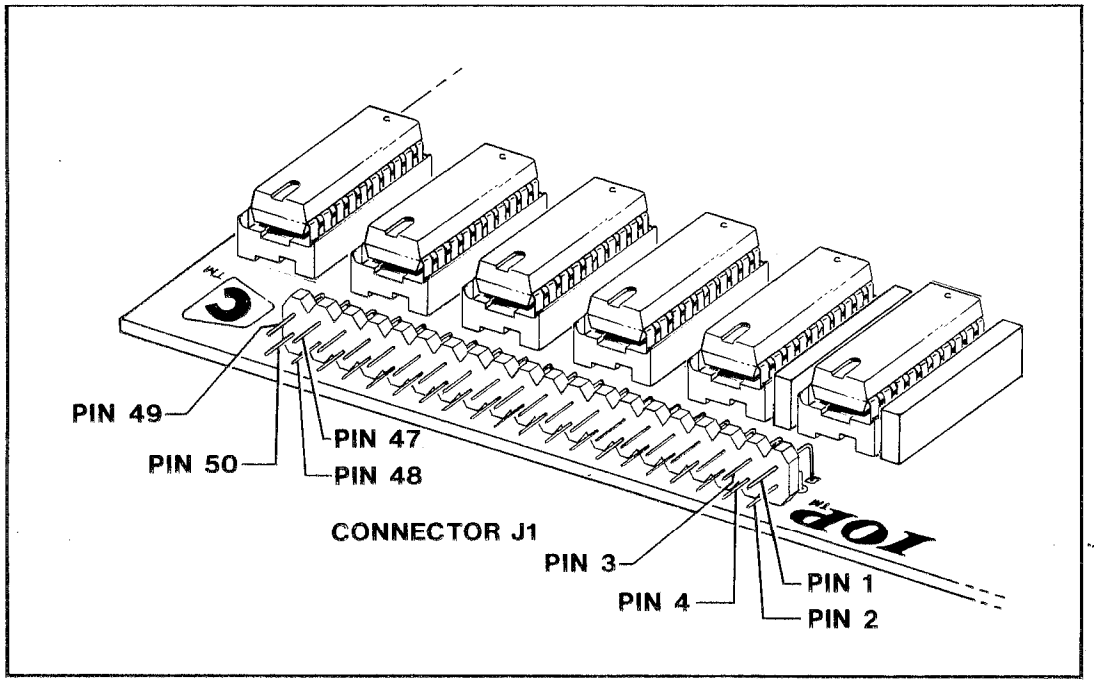


Figure 18: IOP CONNECTOR J1 PIN NUMBERS

2.9 IOP/HOST COMMUNICATION

This section illustrates with the help of a few simple programming examples, how the host and IOP exchange data, commands, and status. Both IOP and host interrupts are assumed disabled in all examples; interrupts are treated fully below in Chapter 3. Refer to Appendix A for complete descriptions of all IOP register bits used in the programming examples, and Cromemco's IOP Development Software Instruction Manual for instructions on creating, downloading, debugging and executing the programs. The first example program shows how the host sends data to the IOP.

DATA: HOST -> IOP

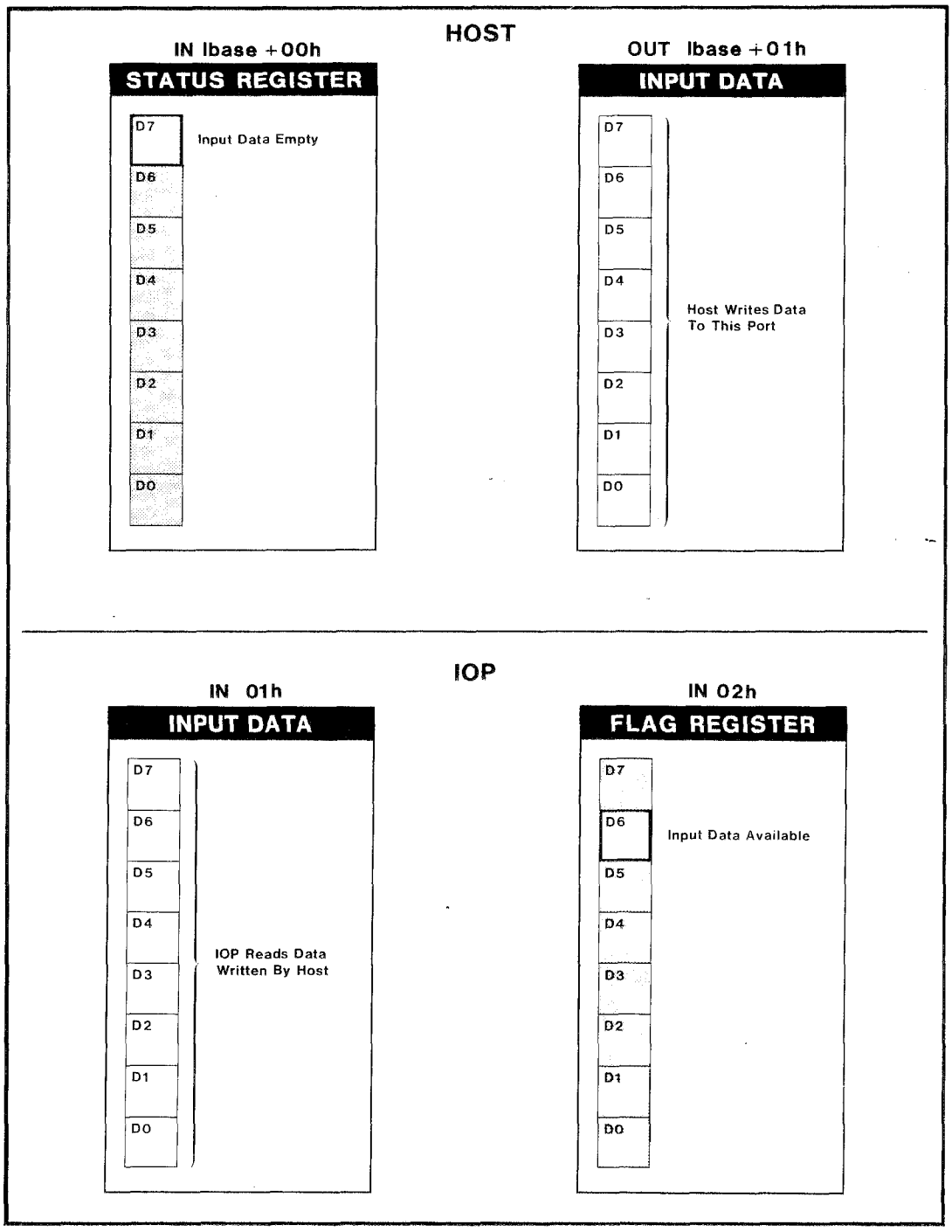


Figure 19: HOST DATA TO IOP

Figure 19 shows the IOP register bits used in the data transfer. Two program segments appear below: one resides in host memory beginning at 100h, the beginning of the user program area assuming a CDOS operating system; and the second resides in IOP memory beginning at 4000h, the start of IOP RAM assuming the standard memory map. The host program sends the contents of RAM buffer HBUFF0 to the IOP by polling **Status Register** bit **Input Data Empty** until set, then outputting a single character to register **Input Data**, then polling bit **Input Data Empty** again, sending another character, and so on. The IOP program receives each character by polling **Flag Register** bit **Input Data Available** until set, then inputting a single character from register **Input Data**, then polling bit **Input Data Available** again, reading another character, and so on. Each character read by the IOP is stored sequentially in IOP RAM buffer IBUFF0.

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       HOST PROGRAM 1
                        0003
                        0004  ;  Host To IOP Data Transfer
                        0005
      (00CE)            0006  IBASE   EQU     0CEh            ;IOP Base Address.
      (00CE)            0007  STATUS  EQU     IBASE+00h       ;Status Register.
      (00CF)            0008  INDATA  EQU     IBASE+01h       ;Input Data Register.
      (0007)            0009  IDE     EQU     7               ;Input Data Empty bit D7.
                        0010
      (0100)            0011          ORG     100h            ;Start of CDOS user area.
                        0012
                        0013          ;=======================
                        0014
0100  211401            0015  HOST1:  LD      HL,HBUFF0       ;Point to host buffer.
                        0016
0103  DBCE              0017  IDE?:   IN      A,(STATUS)      ;Loop until bit Input
0105  CB7F              0018          BIT     IDE,A           ;Data Empty is set...
0107  28FA              0019          JR      Z,IDE?          ;
                        0020
0109  7E                0021          LD      A,(HL)          ;then get a buffer character
010A  D3CF              0022          OUT     (INDATA),A      ;and send to IOP.
                        0023
010C  FE24              0024          CP      '$'             ;Was the character a '$'
010E  2803              0025          JR      Z,DONE          ;delimiter?  Exit if so,
                        0026
0110  23                0027          INC     HL              ;otherwise loop back and
0111  18F0              0028          JR      IDE?            ;send the next character.
                        0029
0113  76                0030  DONE:   HALT                    ;
                        0031
                        0032          ;=======================
                        0033
0114  61626331          0034  HBUFF0: DB      'abc123$'       ;Host buffer data.
                        0035
011B  (0100)            0036          END     HOST1
```

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       IOP PROGRAM 1
                        0003
                        0004  ;  Host To IOP Data Transfer
                        0005
      (0001)            0006  INDATA  EQU     01h             ;Input Data Register.
      (0002)            0007  FLAGS   EQU     02h             ;Flag Register.
      (0006)            0008  IDA     EQU     6               ;Input Data Available bit D6.
                        0009
      (4000)            0010          ORG     4000h           ;Start of IOP RAM memory.
                        0011
                        0012          ;=======================
                        0013
4000  211440            0014  IOP1:   LD      HL,IBUFF0       ;Point to IOP buffer.
                        0015
4003  DB02              0016  IDA?:   IN      A,(FLAGS)       ;Loop until bit Input
4005  CB77              0017          BIT     IDA,A           ;Data Available is set...
4007  28FA              0018          JR      Z,IDA?          ;
                        0019
4009  DB01              0020          IN      A,(INDATA)      ;then get a character from
400B  77                0021          LD      (HL),A          ;the host and stash in buffer.
                        0022
400C  FE24              0023          CP      '$'             ;Was the character a '$'
400E  2803              0024          JR      Z,DONE          ;delimiter?  Exit if so,
                        0025
4010  23                0026          INC     HL              ;otherwise loop back to get
4011  18F0              0027          JR      IDA?            ;and store another character.
                        0028
4013  76                0029  DONE:   HALT                    ;
                        0030
                        0031          ;=======================
                        0032
4014  (0100)            0033  IBUFF0: DS      100h            ;IOP buffer data.
                        0034
4114  (4000)            0035          END     IOP1
```

DATA: HOST <- IOP

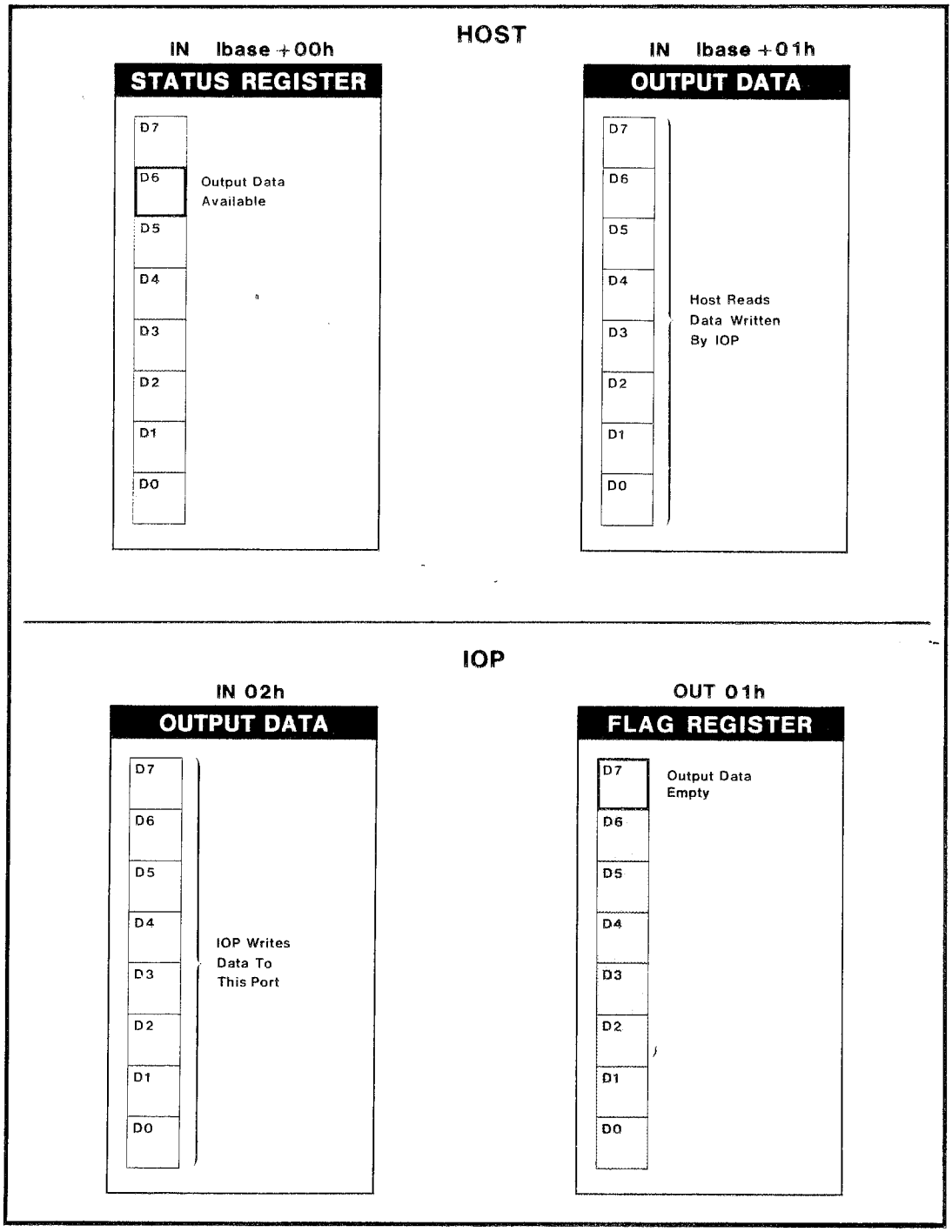


Figure 20: IOP DATA TO HOST

Figure 20 shows the IOP register bits used to transfer data from the IOP back to the host. Again two program segments appear below, one residing in host memory and the other in IOP memory. The IOP program sends the contents of RAM buffer IBUFF1 to the host by polling **Flag Register** bit **Output Data Empty** until set, then outputting a single character to register **Output Data**, then polling bit **Output Data Empty** again, sending another character, and so on. The host program receives each character by polling **Status Register** bit **Output Data Available** until set, then inputting a single character from register **Output Data**, then polling bit **Output Data Available** again, reading another character, and so on. Each character read by the host is stored sequentially in host RAM buffer HBUFF1.

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       HOST PROGRAM 2
                        0003
                        0004  ;  IOP To Host Data Transfer
                        0005
      (00CE)            0006  IBASE   EQU     0CEh            ;IOP Base Address.
      (00CE)            0007  STATUS  EQU     IBASE+00h       ;Status Register.
      (00CF)            0008  OUTDATA EQU     IBASE+01h       ;Output Data Register.
      (0006)            0009  ODA     EQU     6               ;Output Data Available bit D6.
                        0010
      (0100)            0011          ORG     100h            ;Start of CDOS user area.
                        0012
                        0013          ;=======================
                        0014
0100  211401            0015  HOST2:  LD      HL,HBUFF1       ;Point to host buffer.
                        0016
0103  DBCE              0017  ODA?:   IN      A,(STATUS)      ;Loop until bit Output
0105  CB77              0018          BIT     ODA,A           ;Data Available is set...
0107  28FA              0019          JR      Z,ODA?          ;
                        0020
0109  DBCF              0021          IN      A,(OUTDATA)     ;then get a character from
010B  77                0022          LD      (HL),A          ;the IOP and stash in buffer.
                        0023
010C  FE24              0024          CP      '$'             ;Was the character a '$'
010E  2803              0025          JR      Z,DONE          ;delimiter?  Exit if so,
                        0026
0110  23                0027          INC     HL              ;otherwise loop back to get
0111  18F0              0028          JR      ODA?            ;and store another character.
                        0029
0113  76                0030  DONE:   HALT                    ;
                        0031
                        0032          ;=======================
                        0033
0114  (0100)            0034  HBUFF1: DS      100h            ;Host buffer data.
                        0035
0214  (0100)            0036          END     HOST2
```

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       IOP PROGRAM 2
                        0003
                        0004  ;  IOP to Host Data Transfer
                        0005
      (0001)            0006  OUTDATA EQU     01h             ;Output Data Register.
      (0002)            0007  FLAGS   EQU     02h             ;Flag Register.
      (0007)            0008  ODE     EQU     7               ;Output Data Empty bit D7.
                        0009
      (4000)            0010          ORG     4000h           ;Start of IOP RAM memory.
                        0011
                        0012          ;=======================
                        0013
4000  211440            0014  IOP2:   LD      HL,IBUFF1       ;Point to IOP buffer.
                        0015
4003  DB02              0016  ODE?:   IN      A,(FLAGS)       ;Loop until bit Output
4005  CB7F              0017          BIT     ODE,A           ;Data Empty is set...
4007  28FA              0018          JR      Z,ODE?          ;
                        0019
4009  7E                0020          LD      A,(HL)          ;then get a buffer character
400A  D301              0021          OUT     (OUTDATA),A     ;and send to host.
                        0022
400C  FE24              0023          CP      '$'             ;Was the character a '$'
400E  2803              0024          JR      Z,DONE          ;delimiter?  Exit if so,
                        0025
4010  23                0026          INC     HL              ;otherwise loop back and
4011  18F0              0027          JR      ODE?            ;send the next character.
                        0028
4013  76                0029  DONE:   HALT                    ;
                        0030
                        0031          ;=======================
                        0032
4014  61626331          0033  IBUFF1: DB      'abc123$'       ;IOP buffer data.
                        0034
401B  (4000)            0035          END     IOP2
```

COMMANDS: HOST -> IOP

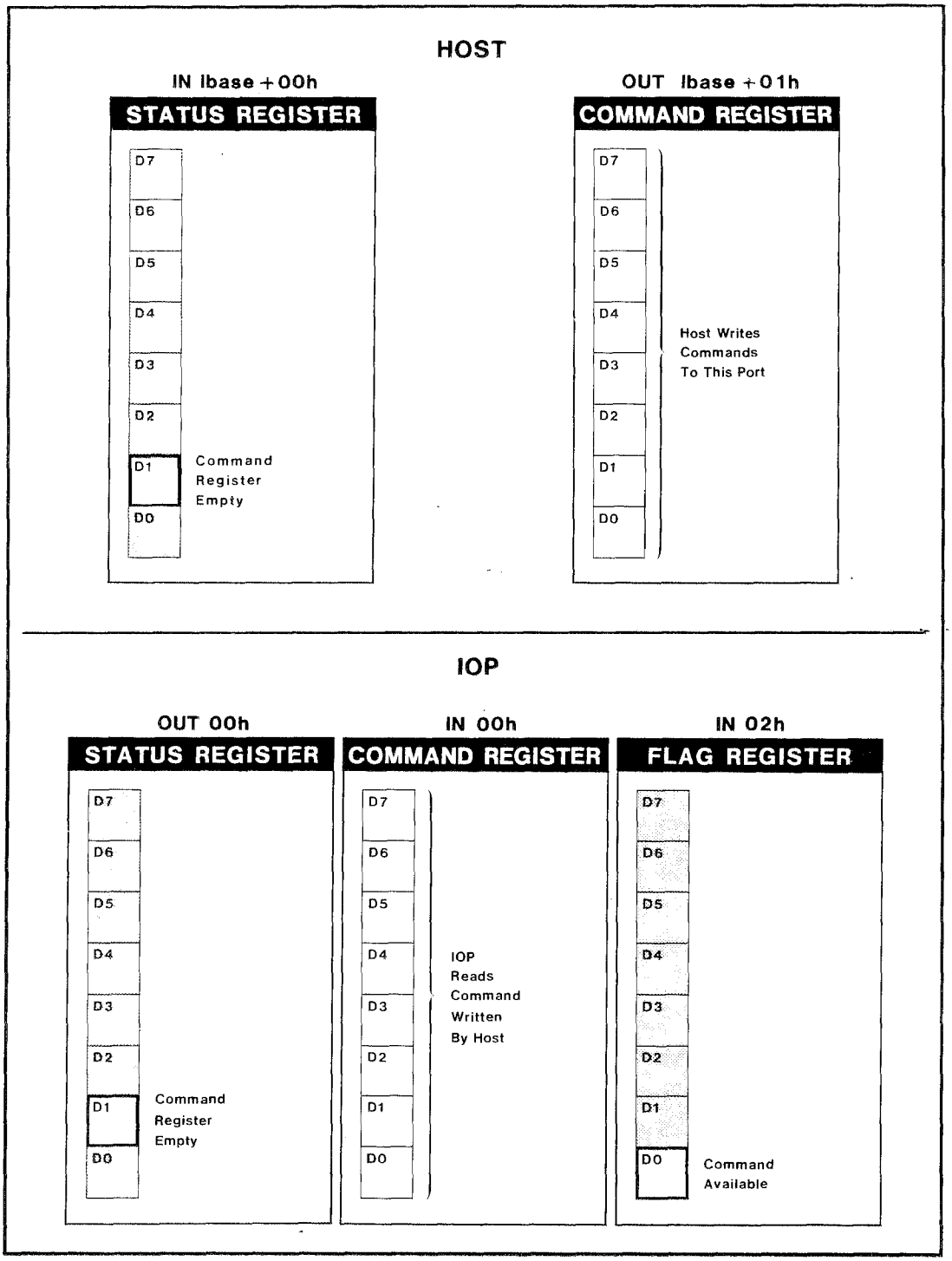


Figure 21: HOST COMMANDS TO IOP

The host passes commands to the IOP in much the same way that it writes data to the IOP; the main difference between the two is that bit **Command Register Empty** is IOP software controlled (the bit is **not** reset when the IOP reads the **Command Register** -- see Appendix A). Command interpretation is completely software defined. In the program segments which follow, host command byte 00h causes the IOP to reset its **Control Register** bit **Enable S-100 Interrupt Requests** (a task the host CPU cannot directly accomplish itself), and command byte 0FFh causes the IOP to set the same bit. Any other command byte is considered a null operation by the IOP program.

```
CROMEMCO Z80 Macro Assembler version 03.07

                    0001
                    0002  ;       HOST PROGRAM 3
                    0003
                    0004  ;  Host To IOP Command Transfer
                    0005
     (00CE)         0006  IBASE   EQU     0CEh            ;IOP Base Address.
     (00CE)         0007  STATUS  EQU     IBASE+00h       ;Status Register.
     (00CE)         0008  COMMAND EQU     IBASE+00h       ;Command Register.
     (0001)         0009  CRE     EQU     1               ;Command Register Empty bit D1.
     (0000)         0010  COM0    EQU     00h             ;Disable Interrupts Command.
     (00FF)         0011  COM1    EQU     0FFh            ;Enable Interrupts Command.
                    0012
     (0100)         0013          ORG     100h            ;Start of CDOS user area.
                    0014
                    0015          ;=======================
                    0016
0100  310010        0017  HOST3:  LD      SP,1000h        ;Locate stack.
                    0018
0103  CD1201        0019          CALL    CRE?            ;Send command to disable
0106  3E00          0020          LD      A,COM0          ;S-100 interrupts when bit
0108  D3CE          0021          OUT     (COMMAND),A     ;Command Register Empty is set,
                    0022
010A  CD1201        0023          CALL    CRE?            ;then send command to enable
010D  3EFF          0024          LD      A,COM1          ;S-100 interrupts when bit Command
010F  D3CE          0025          OUT     (COMMAND),A     ;Register Empty is set again.
                    0026
0111  76            0027          HALT                    ;
                    0028
                    0029          ;=======================
                    0030          ; Subroutine CRE?
                    0031          ;=======================
                    0032
0112  DBCE          0033  CRE?:   IN      A,(STATUS)      ;Loop until bit
0114  CB4F          0034          BIT     CRE,A           ;Command Register Empty
0116  28FA          0035          JR      Z,CRE?          ;is set,
0118  C9            0036          RET                     ;then return.
                    0037
                    0038          ;=======================
                    0039
0119  (0100)        0040          END     HOST3
```

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       IOP PROGRAM 3
                        0003
                        0004  ;  Host To IOP Command Transfer
                        0005
     (0000)             0006  COMMAND EQU     00h             ;Command Register.
     (0000)             0007  STATUS  EQU     00h             ;Status Register.
     (0002)             0008  CONTROL EQU     02h             ;Control Register.
     (0002)             0009  FLAGS   EQU     02h             ;Flag Register.
     (0000)             0010  CRA     EQU     0               ;Command Register Available bit D0.
     (0000)             0011  COM0    EQU     00h             ;Disable Interrupts Command.
     (00FF)             0012  COM1    EQU     0FFh            ;Enable Interrupts Command.
                        0013
     (4000)             0014          ORG     4000h           ;Start of IOP RAM memory.
                        0015
                        0016          ;=======================
                        0017
                        0018  IOP3:
                        0019
4000 DB02               0020  CRA?:   IN      A,(FLAGS)       ;Loop until bit
4002 CB47               0021          BIT     CRA,A           ;Command Register Available
4004 28FA               0022          JR      Z,CRA?          ;is set...
                        0023
4006 DB00               0024          IN      A,(COMMAND)     ;then get command from host.
                        0025
4008 FE00               0026          CP      COM0            ;Disable S-100
400A 2006               0027          JR      NZ,COM1?        ;interrupts if the
400C 3E00               0028          LD      A,00000000b     ;command byte equals
400E D302               0029          OUT     (CONTROL),A     ;00h,
4010 1808               0030          JR      RESET           ;
                        0031
4012 FEFF               0032  COM1?:  CP      COM1            ;or enable S-100
4014 20EA               0033          JR      NZ,IOP3         ;interrupts if the
4016 3E01               0034          LD      A,00000001b     ;command byte equals
4018 D302               0035          OUT     (CONTROL),A     ;0FFh; do nothing otherwise.
                        0036
401A 3E02               0037  RESET:  LD      A,00000010b     ;Reset bits Command Register Empty
401C D300               0038          OUT     (STATUS),A      ;and Command Register Available by
                        0039                                  ;writing a logic 1 to D1 (must be
                        0040                                  ;done by IOP software).
                        0041
401E 18E0               0042          JR      IOP3            ;
                        0043
                        0044          ;=======================
                        0045
4020 (4000)             0046          END     IOP3
```

STATUS: HOST <- IOP

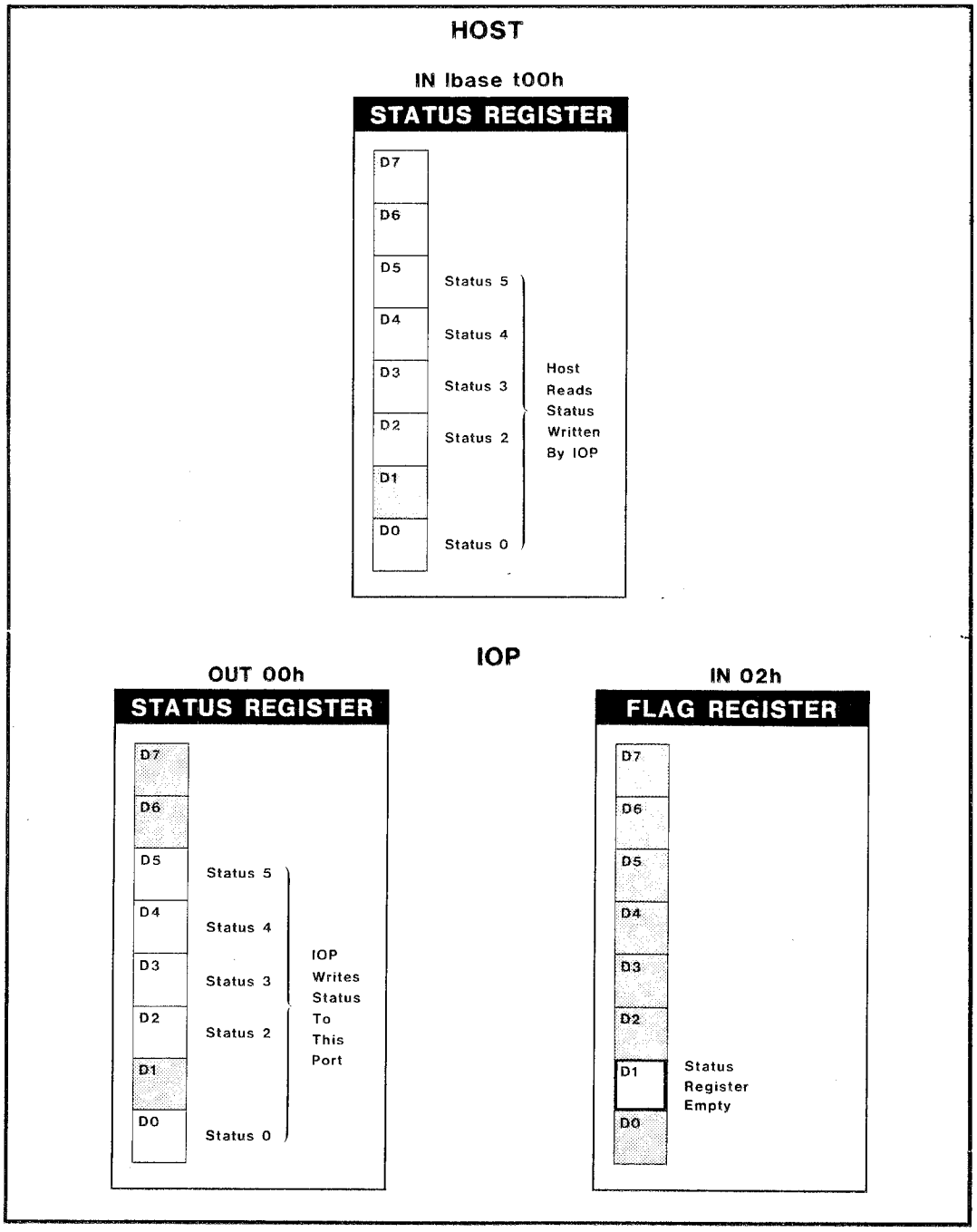


Figure 22: IOP STATUS TO HOST

The IOP writes software defined status to the host by outputting a byte to the **Status Register** when **Flag Register** bit **Status Register Empty** is set. Only bits D5 through D2 and D0 are changed to match the byte output by the IOP; bits D7, D6 and D1 reflect hardware controlled IOP status. Note that the host must poll the **Status Register** itself to determine if new software defined IOP status is available since no host handshake line is provided for this function (the IOP software could easily make one of the five status bits serve as a handshake line, however). In the program segments which follow, the host sends the IOP three pairs of unsigned binary numbers to multiply by repeated addition. The IOP sends each computation back to the host with **Status Register** bit **Status 0** reset if no eight bit overflow occurs on the product, or the product modulus 256 with **Status 0** set if an overflow does occur.

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       HOST PROGRAM 4
                        0003
                        0004  ;  IOP To Host Status Transfer
                        0005
     (00CE)             0006  IBASE   EQU     0CEh            ;IOP Base Address.
     (00CE)             0007  STATUS  EQU     IBASE+00h       ;Status Register.
     (00CF)             0008  OUTDATA EQU     IBASE+01h       ;Output Data Register.
     (0006)             0009  ODA     EQU     6               ;Output Data Available bit D6.
     (00CF)             0010  INDATA  EQU     IBASE+01h       ;Input Data Register.
     (0007)             0011  IDE     EQU     7               ;Input Data Empty bit D7.
                        0012
     (0100)             0013          ORG     100h            ;Start of CDOS user area.
                        0014
                        0015          ;=======================
                        0016
0100 310010             0017  HOST4:  LD      SP,1000h        ;Locate stack.
                        0018
0103 213B01             0019          LD      HL,FACT1        ;Point to first two factors
0106 CD1601             0020          CALL    MULTPY          ;and multiply.
0109 213F01             0021          LD      HL,FACT2        ;Point to second two factors
010C CD1601             0022          CALL    MULTPY          ;and multiply.
010F 214301             0023          LD      HL,FACT3        ;Point to third two factors
0112 CD1601             0024          CALL    MULTPY          ;and multiply.
                        0025
0115 76                 0026          HALT                    ;
                        0027
                        0028          ;=======================
                        0029          ; Subroutine MULTPY
                        0030          ;=======================
                        0031
0116 CD3401             0032  MULTPY: CALL    IDE?            ;For each factor,
0119 7E                 0033          LD      A,(HL)          ;wait until bit
011A D3CF               0034          OUT     (INDATA),A      ;Input Data Empty is set,
011C 23                 0035          INC     HL              ;then pass the factor
011D CD3401             0036          CALL    IDE?            ;to the IOP.
0120 7E                 0037          LD      A,(HL)          ;
0121 D3CF               0038          OUT     (INDATA),A      ;
                        0039
0123 23                 0040          INC     HL              ;
0124 DBCE               0041  ODA?:   IN      A,(STATUS)      ;Wait until bit
0126 CB77               0042          BIT     ODA,A           ;Output Data Available
0128 28FA               0043          JR      Z,ODA?          ;is set,
012A DBCE               0044          IN      A,(STATUS)      ;then read Status Register
012C E601               0045          AND     00000001b       ;and mask off bit Status 0,
012E 77                 0046          LD      (HL),A          ;store at OVFLO,
012F 23                 0047          INC     HL              ;
0130 DBCF               0048          IN      A,(OUTDATA)     ;read product byte and
0132 77                 0049          LD      (HL),A          ;store at PRDCT.
                        0050
0133 C9                 0051          RET                     ;
                        0052
                        0053          ;=======================
                        0054          ; Subroutine IDE?
                        0055          ;=======================
                        0056
0134 DBCE               0057  IDE?:   IN      A,(STATUS)      ;Loop until bit
0136 CB7F               0058          BIT     IDE,A           ;Input Data Empty
0138 28FA               0059          JR      Z,IDE?          ;is set,
013A C9                 0060          RET                     ;then return.
                        0061
                        0062          ;=======================
                        0063
013B 0D07               0064  FACT1:  DB      13,7            ;First two factors.
013D (0001)             0065  OVFLO1: DS      1               ;00h result (no overflow),
013E (0001)             0066  PRDCT1: DS      1               ;5Bh result.
                        0067
013F 0F1C               0068  FACT2:  DB      15,28           ;Second two factors.
0141 (0001)             0069  OVFLO2: DS      1               ;01h result (overflow),
0142 (0001)             0070  PRDCT2: DS      1               ;A4h result (1A4h mod 100h).
                        0071
0143 00EC               0072  FACT3:  DB      0,236           ;Third two factors.
0145 (0001)             0073  OVFLO3: DS      1               ;00h result (no overflow),
0146 (0001)             0074  PRDCT3: DS      1               ;00h result.
                        0075
                        0076          ;=======================
                        0077
0147 (0100)             0078          END     HOST4
```

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       IOP PROGRAM 4
                        0003
                        0004  ;  IOP to Host Status Transfer
                        0005
      (0000)            0006  STATUS  EQU     00h             ;Status Register.
      (0001)            0007  INDATA  EQU     01h             ;Input Data Register.
      (0001)            0008  OUTDATA EQU     01h             ;Output Data Register.
      (0002)            0009  FLAGS   EQU     02h             ;Flag Register.
      (0001)            0010  SRE     EQU     1               ;Status Register Empty bit D1.
      (0006)            0011  IDA     EQU     6               ;Input Data Available bit D6.
      (0007)            0012  ODE     EQU     7               ;Output Data Empty bit D7.
                        0013
      (4000)            0014          ORG     4000h           ;Start of IOP RAM memory.
                        0015
                        0016          ;=======================
                        0017
4000  310060            0018  IOP4:   LD      SP,6000h        ;Locate stack.
4003  1600              0019          LD      D,00000000b     ;Reset Overflow Flag.
                        0020
4005  CD3340            0021          CALL    IDA?            ;
4008  DB01              0022          IN      A,(INDATA)      ;Get a factor from the host,
400A  B7                0023          OR      A               ;exit with no overflow and
400B  2811              0024          JR      Z,DONE          ;zero product if factor = 00h,
400D  47                0025          LD      B,A             ;otherwise store in Reg. B.
                        0026
400E  CD3340            0027          CALL    IDA?            ;
4011  DB01              0028          IN      A,(INDATA)      ;Get second factor from the host
4013  4F                0029          LD      C,A             ;and store in Reg. C.
                        0030
4014  05                0031  NXTADD: DEC     B               ;Add (C) to itself (B) times.
4015  2807              0032          JR      Z,DONE          ;
4017  81                0033          ADD     C               ;
4018  30FA              0034          JR      NC,NXTADD       ;
401A  1601              0035          LD      D,00000001b     ;If carry, set Overflow Flag.
401C  18F6              0036          JR      NXTADD          ;
                        0037
401E  F5                0038  DONE:   PUSH    AF              ;Save product on stack.
                        0039
401F  DB02              0040  SRE?:   IN      A,(FLAGS)       ;When bit
4021  CB4F              0041          BIT     SRE,A           ;Status Register Empty is set,
4023  28FA              0042          JR      Z,SRE?          ;
4025  7A                0043          LD      A,D             ;load Overflow Flag into bit
4026  D300              0044          OUT     (STATUS),A      ;Status 0 and write to host.
                        0045
4028  DB02              0046  ODE?:   IN      A,(FLAGS)       ;When bit
402A  CB7F              0047          BIT     ODE,A           ;Output Data Empty is set,
402C  28FA              0048          JR      Z,ODE?          ;
402E  F1                0049          POP     AF              ;pop product from stack and
402F  D301              0050          OUT     (OUTDATA),A     ;write to Output Data Register.
                        0051
4031  18CD              0052          JR      IOP4            ;Repeat.
                        0053
                        0054          ;=======================
                        0055          ; Subroutine IDA?
                        0056          ;=======================
                        0057
4033  DB02              0058  IDA?:   IN      A,(FLAGS)       ;Loop until bit
4035  CB77              0059          BIT     IDA,A           ;Input Data Available
4037  28FA              0060          JR      Z,IDA?          ;is set,
4039  C9                0061          RET                     ;then return.
                        0062
                        0063          ;=======================
                        0064
403A  (4000)            0065          END     IOP4
```

The last two program segments of this section illustrate data, command and status transfers combined in one inconsequential task. The host CPU begins by sending the ASCII string of characters **Time flies like an arrow,** to the IOP after observing **Status Register** bit **Status 0** set. The IOP concurrently begins by resetting bit **Status 0,** then later setting it after a software delay loop (about .5 Sec) times out. These two actions together imply that the host must wait until the delay loop times out before passing the message to the IOP. After receiving the message, the IOP then appends **fruit flies like an apple.<CR><LF>** to the end of the previous message and sends it back to the host after the host has written a command byte (any value) to the **Command Register.** After sending **Time flies like an arrow,** to the IOP, the host also goes through a software delay loop (again about .5 Sec) before issuing the command which starts the message back. The host sends each message character it receives from the IOP to the system console for viewing. After the message is completely received by the host, the entire process is repeated. The net effect of the programs is then to cause the message **Time flies like an arrow, fruit flies like an apple.<CR><LF>** to repeatedly appear on the system console with a pause equal to the sum of the two software delay loops (about 1 Sec) separating each message.

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       HOST PROGRAM 5
                        0003
                        0004  ;  Host -> IOP -> Host Message Transfer
                        0005
      (00CE)            0006  IBASE   EQU     0CEh            ;IOP Base Address.
      (00CE)            0007  COMMAND EQU     IBASE+00h       ;Command Register.
      (00CE)            0008  STATUS  EQU     IBASE+00h       ;Status Register.
      (0003)            0009  STAT3   EQU     3               ;Status 3 bit D3.
      (00CF)            0010  OUTDATA EQU     IBASE+01h       ;Output Data Register.
      (0006)            0011  ODA     EQU     6               ;Output Data Available bit D6.
      (00CF)            0012  INDATA  EQU     IBASE+01h       ;Input Data Register.
      (0007)            0013  IDE     EQU     7               ;Input Data Empty bit D7.
      (0005)            0014  CDOS    EQU     0005h           ;CDOS Call Address.
      (0002)            0015  WRTCON  EQU     2               ;Write Console CDOS call.
                        0016
      (0100)            0017          ORG     100h            ;Start of CDOS user area.
                        0018
                        0019          ;=======================
                        0020
0100  310010            0021  HOST5:  LD      SP,1000h        ;Locate stack.
0103  213901            0022          LD      HL,MESSG        ;Point to beginning of message.
                        0023
0106  DBCE              0024  STAT3?: IN      A,(STATUS)      ;Wait for the IOP to
0108  CB5F              0025          BIT     STAT3,A         ;set bit Status 3,
010A  28FA              0026          JR      Z,STAT3?        ;
                        0027
010C  DBCE              0028  IDE?:   IN      A,(STATUS)      ;then send all message
010E  CB7F              0029          BIT     IDE,A           ;characters to the IOP
0110  28FA              0030          JR      Z,IDE?          ;up to and including the
0112  7E                0031          LD      A,(HL)          ;'$' delimiter.
0113  23                0032          INC     HL              ;
0114  D3CF              0033          OUT     (INDATA),A      ;
0116  FE24              0034          CP      '$'             ;
0118  20F2              0035          JR      NZ,IDE?         ;
                        0036
011A  210000            0037          LD      HL,0            ;Then enter a software
011D  2B                0038  PAUSE:  DEC     HL              ;wait loop while the
011E  7C                0039          LD      A,H             ;IOP waits for this
011F  B5                0040          OR      L               ;program to issue any
0120  20FB              0041          JR      NZ,PAUSE        ;command.
                        0042
0122  0E02              0043          LD      C,WRTCON        ;Set up for CDOS console writes.
0124  D3CE              0044          OUT     (COMMAND),A     ;Issue a command (00h) to IOP.
                        0045
0126  DBCE              0046  ODA?:   IN      A,(STATUS)      ;The IOP now begins to send
0128  CB77              0047          BIT     ODA,A           ;back updated message.
012A  28FA              0048          JR      Z,ODA?          ;
012C  DBCF              0049          IN      A,(OUTDATA)     ;Read each message characterr
012E  5F                0050          LD      E,A             ;and send it to the system
012F  FE24              0051          CP      '$'             ;console for veiwing
0131  CA0001      R     0052          JP      Z,HOST5         ;(exclude '$' delimiter),
0134  CD0500            0053          CALL    CDOS            ;then start all over.
0137  18ED              0054          JR      ODA?            ;
                        0055
                        0056          ;=======================
                        0057
0139  54696D65          0058  MESSG:  DB 'Time flies like an arrow,$'
                        0059
0153  (0100)            0060          END     HOST5
```

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       IOP PROGRAM 5
                        0003
                        0004  ;  Host -> IOP -> Host Message Transfer
                        0005
        (0000)          0006  STATUS  EQU     00h             ;Status Register.
        (0001)          0007  INDATA  EQU     01h             ;Input Data Register.
        (0001)          0008  OUTDATA EQU     01h             ;Output Data Register.
        (0002)          0009  FLAGS   EQU     02h             ;Flag Register.
        (0000)          0010  CRA     EQU     0               ;Command Register Available bit D0.
        (0001)          0011  SRE     EQU     1               ;Status Register Empty bit D1.
        (0006)          0012  IDA     EQU     6               ;Input Data Available bit D6.
        (0007)          0013  ODE     EQU     7               ;Output Data Empty bit D7.
        (000D)          0014  CR      EQU     0Dh             ;ASCII Carriage Return.
        (000A)          0015  LF      EQU     0Ah             ;ASCII Line Feed.
                        0016
        (4000)          0017          ORG     4000h           ;Start of IOP RAM memory.
                        0018
                        0019          ;======================
                        0020
4000    310060          0021  IOP5:   LD      SP,6000h        ;Locate stack.
                        0022
4003    3E00            0023          LD      A,00000000b     ;Reset bit Status 3.
4005    D300            0024          OUT     (STATUS),A      ;
                        0025
4007    210000          0026          LD      HL,0            ;Host now waiting
400A    2B              0027  PAUSE:  DEC     HL              ;while this software
400B    7C              0028          LD      A,H             ;wait loop times out
400C    B5              0029          OR      L               ;before bit Status 3
400D    20FB            0030          JR      NZ,PAUSE        ;is set.
                        0031
400F    3E08            0032          LD      A,00001000b     ;Set bit Status 3; prompt
4011    D300            0033          OUT     (STATUS),A      ;Host to send message.
                        0034
4013    215340          0035          LD      HL,INBUFF       ;Store each message
4016    DB02            0036  IDA?    IN      A,(FLAGS)       ;character starting
4018    CB77            0037          BIT     IDA,A           ;at INBUFF up to
401A    28FA            0038          JR      Z,IDA?          ;and including '$'
401C    DB01            0039          IN      A,(INDATA)      ;delimiter.
401E    77              0040          LD      (HL),A          ;
401F    23              0041          INC     HL              ;
4020    FE24            0042          CP      '$'             ;
4022    20F2            0043          JR      NZ,IDA?         ;
                        0044
4024    DB02            0045  CRA?:   IN      A,(FLAGS)       ;IOP now waits for host
4026    CB47            0046          BIT     CRA,A           ;to write any command
4028    28FA            0047          JR      Z,CRA?          ;to Command Register.
                        0048
402A    3E02            0049          LD      A,00000010b     ;Reset bits Command Register Empty
402C    D300            0050          OUT     (STATUS),A      ;and Command Register Available by
                        0051                                  ;writing a logic 1 to D1 (must be
                        0052                                  ;done by IOP software).
                        0053
402E    0E02            0054          LD      C,2             ;Counter used below.
4030    215340          0055          LD      HL,INBUFF       ;Send back message
4033    CD3E40          0056          CALL    SNDBUF          ;received from host,
4036    215341          0057          LD      HL,OUTBUFF      ;append new message
4039    CD3E40          0058          CALL    SNDBUF          ;and also send to host.
                        0059
403C    18C2            0060          JR      IOP5            ;Repeat process.
                        0061
                        0062          ;======================
                        0063          ; Subroutine SNDBUF
                        0064          ;======================
                        0065
                        0066  SNDBUF:
403E    DB02            0067  ODE?:   IN      A,(FLAGS)       ;Sequentially send
4040    CB7F            0068          BIT     ODE,A           ;each message character
4042    28FA            0069          JR      Z,ODE?          ;when bit Output Data
4044    7E              0070          LD      A,(HL)          ;Empty is set except for
4045    23              0071          INC     HL              ;'$' delimiter.
4046    FE24            0072          CP      '$'             ;If character is the
4048    2804            0073          JR      Z,TESTC         ;'$' delimiter,
404A    D301            0074          OUT     (OUTDATA),A     ;
404C    18F0            0075          JR      ODE?            ;
                        0076
```

```
404E  0D          0077  TESTC:  DEC     C               ;then do not send it
404F  C0          0078          RET     NZ              ;at the end of INBUFF,
4050  D301        0079          OUT     (OUTDATA),A     ;but do send it at
4052  C9          0080          RET                     ;the end of OUTBUFF
                  0081                                  ;(at the end of
                  0082                                  ; entire message).
                  0083
                  0084          ;=======================
                  0085
4053  (0100)      0086  INBUFF: DS      100h            ;Input buffer for host data.
4153  20667275    0087  OUTBUFF:DB ' fruit flies like'
4164  20616E20    0088          DB ' an apple.',CR,LF,'$'
                  0089
4171  (4000)      0090          END     IOP5
```

2.10 IOP/C-BUS PERIPHERAL COMMUNICATION

The IOP communicates with the C-Bus peripherals which it controls over the 50 conductor C-Bus cable. Since the C-Bus supports a full compliment of Z-80A address, data and control lines (except $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ -- DMA functions are software controlled), the IOP may participate in the full range of Z-80A data transfer modes with the C-Bus peripherals: memory reads and writes; I/O reads and writes; vectored interrupt requests and acknowledgments; and DMA requests and acknowledgments.

The few restrictions which apply to these transfers are listed below:

MEMORY READS/WRITES: When external C-Bus memory addresses overlap on board IOP RAM or ROM memory, then: (1) the IOP Z-80A reads data and executes object code from IOP on board memory, not external memory, and (2) the Z-80A writes data to IOP on board memory and external C-Bus memory in parallel. If C-Bus line $\overline{\text{WAIT}}$ must be controlled to synchronize the Z-80A to slow external memory, it is the C-Bus peripheral's responsibility to do so.

I/O READS/WRITES: When external C-Bus I/O port addresses overlap on board IOP I/O ports (IN 00h - IN 02h, OUT 00h - OUT 03h), then: (1) the IOP Z-80A inputs data from IOP on board IN ports, not external IN ports, and (2) the IOP Z-80A outputs data to IOP on board OUT ports and external OUT ports in parallel. If C-Bus line $\overline{\text{WAIT}}$ must be controlled to synchronize the Z-80A to slow external I/O ports, it is the C-Bus peripheral's responsibility to do so.

INTERRUPT REQUESTS/ACKNOWLEDGMENTS: Multiple IOP interrupt request sources must be prioritized using C-Bus line $\overline{\text{PRI 3}}$, <48>, and optionally $\overline{\text{PRI 2}}$, <47>, and $\overline{\text{PRI 1}}$, <46>. See Section 3.1 for full details.

DMA REQUESTS/ACKNOWLEDGMENTS: DMA requests by C-Bus peripherals are polled by the IOP software; and DMA acknowledgments which cause the C-Bus address, data and control lines to float are also under IOP software control. Consequently, DMA requests from C-Bus peripherals are not necessarily acknowledged quickly, or at all for that matter, depending on the IOP software design. See Section 3.4 for full details.

The IOP user may elect to design and build custom C-Bus interfaces when using the IOP as an intelligent controller. Figure 23 shows an example custom design

which provides 1 Kbyte of 2114 RAM memory spanning FC00h - FFFFh, one LED display port (OUT FFh), and one switch register input port (IN FFh). Note that the peripheral I/O ports do not overlap IOP ports, and assuming the standard IOP memory map, the peripheral RAM does not overlap IOP memory.

Refer to the figure and note that 2114 RAM memory is enabled when the following Boolean expression is true:

CS = (RD + WR) AND (MREQ) AND (FC00h <= ADDRESS <= FFFFh)

This expression guarantees that no data bus conflicts between Z-80A write data and 2114 read data can occur since it is true only while the Z-80A is asserting either $\overline{RD}$ or $\overline{WR}$ active low.

The contents of the switch register are gated onto the C-Bus data bus while the following Boolean expression is true; a closed switch is read as a logic 0, an open switch as logic 1.

GATE = (RD) AND (IORQ) AND (PORT ADDRESS = FFh),

The byte output to the display register is latched when the Boolean expression,

$\overline{CP}$ = (WR) AND (IORQ) AND (PORT ADDRESS = FFh),

changes from true to false on the trailing edge of the $\overline{WR}$ pulse. A logic 0 data bit turns its corresponding LED on, a logic 1 turns it off. All display register bits are reset (all LEDs on) by a POC or C-Bus Clear.

Refer to Section 4.11 which discusses C-Bus signals and waveforms for each Z-80A machine cycle type.

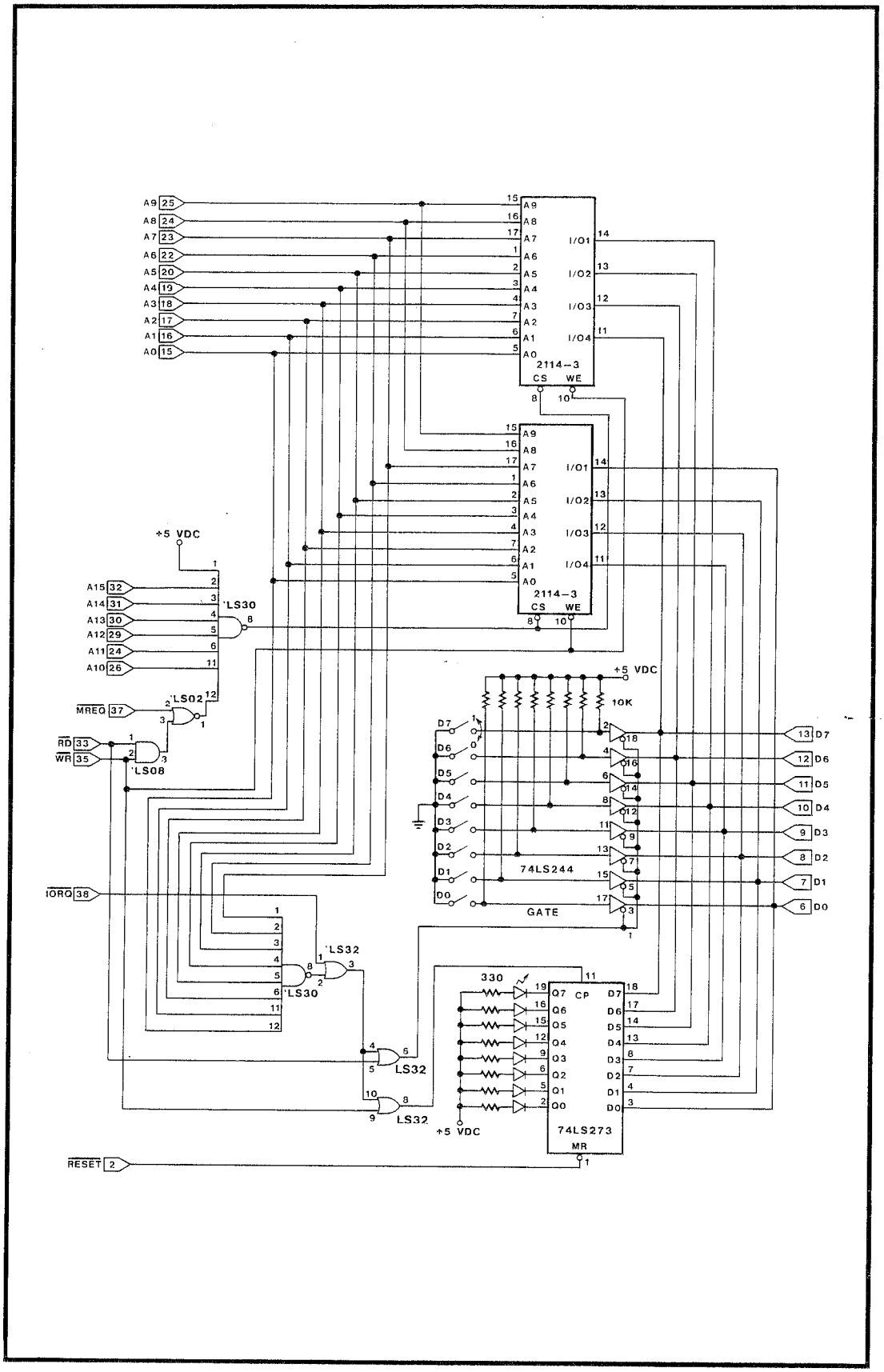


Figure 23: AN EXAMPLE CUSTOM C-BUS PERIPHERAL DESIGN

2.11 IOP SET UP AND INITIAL CHECK OUT

The following initial check out procedure exercises the IOP Z-80A, RAM memory, ROM memory, **Status Register, Output Data** register, **Input Data** register and **Flag Register.** Successfully completing this procedure implies that the major IOP subsystems are functioning properly.

EQUIPMENT SET UP

The following equipment will be needed for this test procedure: a disk based CDOS or Cromix S-100 host system with console; disk program IOPEX.COM, supplied in the **IOP Development Software** package; a Cromemco IOP board with standard memory mapping PROM in socket IC10; and a Cromemco **IOPMON** 2516 EPROM installed in IOP socket ROM0.

1. Set the IOP base address to CEh (the switch setting shown in Figure 12).

2. Install the IOP board in an empty S-100 bus slot while S-100 power is turned off.

3. Turn host S-100 bus power on.

INITIAL CHECK OUT

1. Boot up the Cromix or CDOS operating system.

2. Run program IOPEX.COM by typing on the host console,

   IOPEX

   followed by a **RETURN**

3. Program IOPEX should now sign on with the message

   I/O Processor File Command System - version xx.yy

   and then attempt to establish contact with the running IOP Monitor program, **IOPMON** (**xx** defines the version and **yy** the release number). Once contact is established, IOPEX makes the host console act as the IOP monitor console.

4. The IOPMON program should now sign on with the message

   IOP Monitor xx.yy
   .

   The period character below the sign on banner is the monitor's command prompt.

5. Enter command **Z 4000 S80 33** followed by a **RETURN.** This **IOPMON** command fills (zaps) an 80h long swath of RAM memory starting at address 4000h (the bottom of IOP RAM) with data byte 33h. The command should be echoed exactly as typed.

6. Enter command **DM 4000** followed by a **RETURN.** This command causes **IOPMON** to format and display an 80h (by default) swath of memory starting at IOP RAM address 4000h. You should now see all 33h data.

7. The IOP may be further exercised by entering any other legal **IOPMON** command -- see IOP Development Software Instruction Manual.

8. When finished, type the letters **K** to exit to the operating system (CDOS or Cromix Operating System).

This completes the initial IOP check out. If the procedure is not successful, check each procedure step again. Especially note the following possible problems: IOP base address switch not set to CEh; IOPMON ROM installed in wrong socket, installed backwards, bent under pin, or wrong ROM; IOP not installed properly in the S-100 bus.

Chapter 3

INTERRUPTS AND DMA

This chapter discusses two broad categories of interrupts: **C-Bus Interrupts**, and **S-100 Bus Interrupts.** C-Bus Interrupts refer to interrupt requests issued to, and serviced by the IOP Z-80A. In contrast, S-100 Bus Interrupts refer to interrupt requests issued to, and serviced by the host processor. The last section of this chapter discusses how C-Bus DMA transfers are requested by C-Bus peripherals and acknowledged by the IOP, and how the IOP may participate in DMA transfers on the host S-100 bus.

3.1 C-BUS INTERRUPTS

Non-maskable interrupt requests are issued to the IOP Z-80A by forcing its $\overline{\text{NMI}}$ pin active low; only external peripherals can force this pin low through C-Bus line $\overline{\text{NMI}}$, <45> (see Figure 24). Maskable interrupt requests are issued to the IOP Z-80A by forcing its $\overline{\text{INT}}$ pin active low. Two sources can force this pin low: (1) external peripherals through C-Bus line $\overline{\text{INT}}$, <44>, and (2) host I/O with IOP ports when **Control Register** bit **Enable IOP Interrupts** is set.

NMI Requests And Service Mode

Unlike maskable Z-80A interrupt requests, non-maskable interrupt requests cannot be disabled by a software controlled mask. NMI interrupts are typically used only for crucial system conditions like impending power failures. Any C-Bus peripheral may force line $\overline{\text{NMI}}$ active low, and this falling edge in turn sets a Z-80A internal NMI latch. The following paragraph describes how the Z-80A responds to an NMI interrupt request.

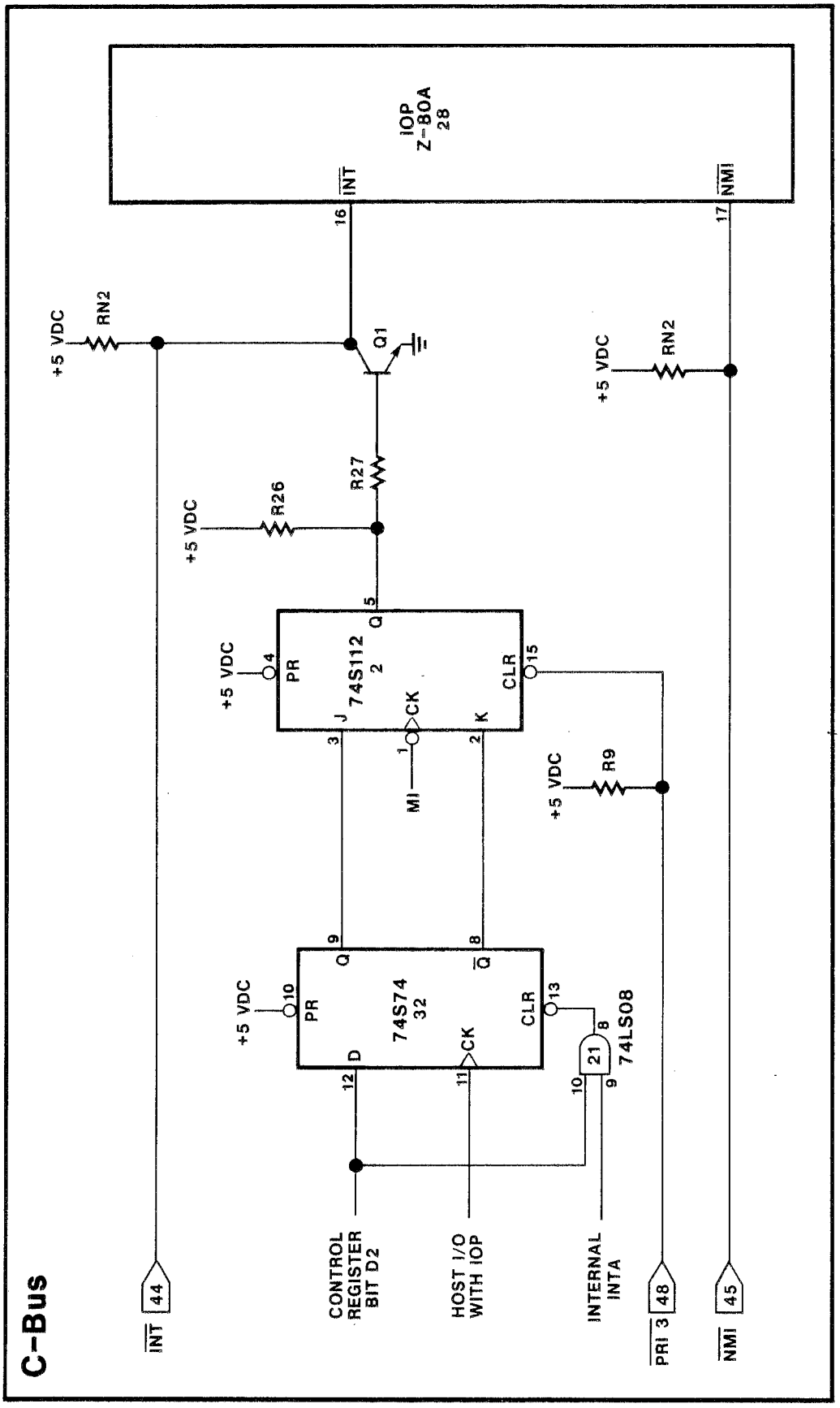


Figure 24: IOP INTERRUPT REQUEST CIRCUITRY

NMI Interrupt Response:

The Z-80A samples its internal NMI latch on the rising edge of the last clock cycle of every instruction executed. If the latch is found set, the Z-80A unconditionally accepts the request by finishing the current instruction, resetting the NMI latch, disabling maskable interrupts, pushing the Program Counter (PC) contents onto the stack, and then jumping to memory address 0066h, the fixed starting address of the NMI service routine (near the beginning of ROM0 assuming the standard IOP memory map). The Z-80A does **not** issue INTA status ($\overline{M1}$ and $\overline{IORQ}$ simultaneously active low) in response to an NMI. A return from the NMI service routine to the interrupted program may be performed by executing a RETN instruction which pops the PC contents from the stack and restores the pre-NMI interrupt mask (maskable interrupts either enabled or disabled), or by executing a RET instruction which pops the PC from the stack but does not restore the pre-NMI interrupt mask (maskable interrupts remain disabled). An NMI interrupt is serviced even when the Z-80A is in the HALTed state; but forcing C-Bus line $\overline{WAIT}$, <5>, active low prevents the current instruction from ending, and thus holds off NMI servicing.

Maskable Interrupt Requests

As mentioned above, maskable interrupt requests are issued to the IOP Z-80A by forcing its $\overline{INT}$ pin active low. Driving the $\overline{INT}$ pin low does not guarantee the Z-80A will acknowledge the request, however, since the $\overline{INT}$ pin may be masked (interrupts disabled) by any one of the following events: (1) a Z-80A reset -- this occurs after a POC or a C-Bus Reset, or (2) a **DI** instruction is executed, or (3) the Z-80A enters an INTA cycle to service any type of interrupt. The $\overline{INT}$ pin is unmasked (interrupts enabled) by any of the following events: (1) an **EI** instruction is executed, or (2) an NMI service routine returns to the background program with a **RETN** instruction if interrupts were enabled when the non-maskable interrupt occurred.

Two sources can force the IOP Z-80A $\overline{INT}$ pin low. The first is C-Bus line $\overline{INT}$, <44>, which is connected directly to the Z-80A. If there are two or more peripherals connected to the C-Bus which may force this

line low, then: (1) their interrupt requests must be prioritized -- see below, and (2) each peripheral must be capable of supplying an interrupt vector during INTA from the IOP unless the Z-80A is operating in Interrupt Mode 1. Typically the Z-80A would be operated in Interrupt Mode 2 (IM2) for efficient interrupt servicing.

The second maskable interrupt source consists of on board IOP circuitry which may be armed to detect a host write to the **Command Register** or the **Input Data** register, or a host read from the **Status Register** or the **Output Data** register. If **Control Register** bit **Enable IOP Interrupts** is set, then this circuitry issues a maskable IOP interrupt request in response. (Maskable interrupt requests resulting from host I/O with IOP ports will be termed **internal IOP interrupts** to contrast them with interrupt requests from external C-Bus peripherals.) This design feature enables the IOP to exchange data, commands, and status with the host on an interrupt driven basis. Interrupt vector FEh is automatically placed on the internal IOP data bus when the Z-80A acknowledges the interrupt request from this source. If **Command Register** bit **Enable IOP Interrupts** is reset, then internal IOP interrupts cannot occur.

C-Bus Interrupt Priorities

The IOP circuitry shown in Figure 24 assigns maskable interrupt requests from C-Bus peripherals higher priority than internal IOP interrupts. To establish these relative priorities, C-Bus peripherals must assert line $\overline{\text{PRI 3}}$, <48>, low in unison with $\overline{\text{INT}}$, <44>, to inhibit internal IOP interrupts. Line $\overline{\text{INT}}$ is typically released during INTA from the IOP when peripheral servicing begins, and $\overline{\text{PRI 3}}$ is released when servicing is completed. Two or more C-Bus peripherals may compete for use of line $\overline{\text{INT}}$, <44>, and in such cases uncommitted C-Bus lines <46> and <47> may be used to establish priorities and coordinate interrupt requests and acknowledgments among the peripherals. Cromemco's C-Bus product line defines <46> to be $\overline{\textbf{PRI 1}}$, and <47> to be $\overline{\textbf{PRI 2}}$; these lines are then used to create a C-Bus interrupt daisy chain as shown in Figure 25.

Here, internal IOP interrupts are assigned the lowest interrupt priority, and External Device 2 is assigned the highest priority (external daisy chain cabling is required to establish priorities among four or more C-Bus peripherals). The $\overline{\text{PRIORITY IN}}$ and $\overline{\text{PRIORITY OUT}}$ lines of each Cromemco C-Bus peripheral behave at the board level exactly like Z-80A family device IEI and IEO

pins of behave at the device level -- see Zilog's application note, **The Z-80A Family Program Interrupt Structure.**

Note the following points when designing a C-Bus interrupt structure using Cromemco C-Bus products connected as shown in Figure 25:

1. If two or more maskable interrupt requests are concurrently pending, then the highest priority request is serviced first.

2. A higher priority interrupt request may be acknowledged while service to a lower priority interrupt is in progress (thus nesting the service routines), but not vice versa.

3. A higher priority interrupt is acknowledged while service to a lower priority interrupt is in progress only if the lower priority service routine has enabled maskable interrupts by executing an **EI** instruction.

4. All interrupt service routines **must** return to the background program with an **RETI** (Return From Interrupt) instruction when two or more C-Bus sources compete for IOP servicing. The RETI opcode (EDh followed by 4Dh) is detected by C-Bus peripherals, and this event is then used to update their $\overline{\text{PRIORITY IN}}$ and $\overline{\text{PRIORITY OUT}}$ lines.

5. All C-Bus priority lines ($\overline{\text{PRI 1}}$ through $\overline{\text{PRI 3}}$) may be left floating if there is only one active maskable C-Bus interrupt source (e.g., only internal IOP interrupts active, or only interrupts from a single Cromemco Quadart active). The RETI instruction should still be used in cases where a single C-Bus peripheral establishes an internal priority structure among two or more Z-80-family devices (such as on a Quadart board).

6. The above points apply to maskable interrupts only; NMI interrupts are unconditionally acknowledged.

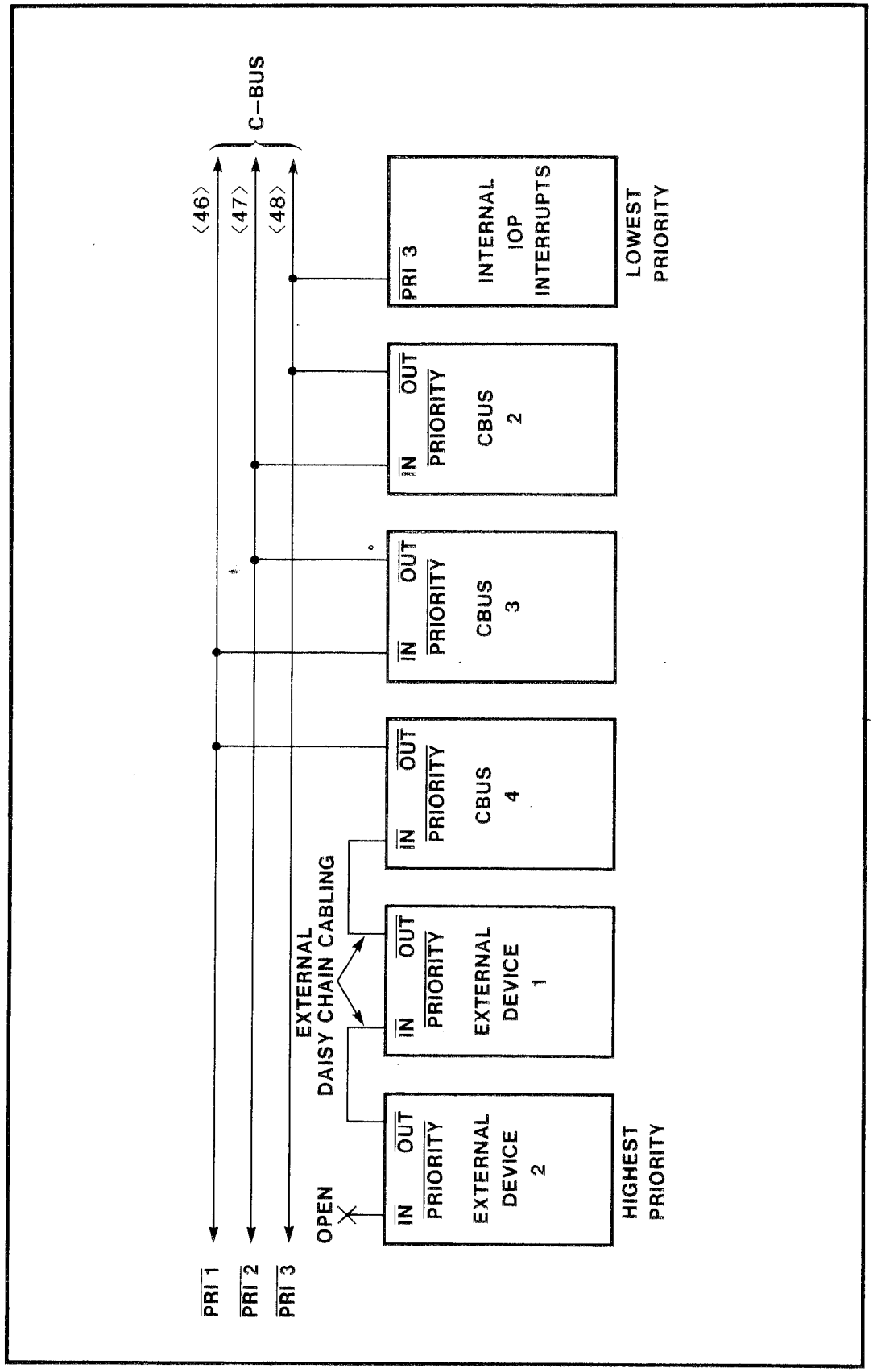


Figure 25: C-BUS PRIORITIES

Maskable Interrupt Service Modes

When maskable interrupt are enabled, the Z-80A samples its $\overline{INT}$ pin on the rising edge of the last clock cycle of every instruction executed (at the same time the internal NMI latch is sampled). If pin $\overline{INT}$ is found active low at this moment, the Z-80A may acknowledge the request in one of three ways, depending on the programmed **Interrupt Mode** -- IM0, IM1 or IM2. Each mode will now be briefly discussed:

Interrupt Mode 0 (IM0)

This mode is selected by default after a POC or an S-100 Bus Reset from the IOP. It is also termed **8080 Mode** since it mimics the 8080 interrupt response, and is software selected by executing an **IM0** instruction. In this mode, the Z-80A finishes the current instruction, disables maskable interrupts, issues INTA status ($\overline{M1}$ and $\overline{IORQ}$ both active low), then fetches the next instruction opcode from the data bus. Meanwhile, the interrupting device monitors the bus for INTA status to gate one of eight **RST** instructions onto the bus as an interrupt service routine vector, and then to remove the interrupt request. **RST 00h** through **RST 38h** are one byte CALL instructions to eight predefined memory addresses. RST 00h is a one byte CALL to 0000h, RST 08h to 0008h, and so on. The Z-80A executes a RST instruction by pushing the PC onto the stack, then jumping to one of the eight addresses defined above. After servicing is complete, an RETI instruction is typically executed which reloads the PC with its original contents from the stack, and program execution resumes from the point of interruption.

Interrupt Mode 1 (IM1)

This mode is selected by executing an IM1 instruction. Here, the Z-80A acknowledges interrupt requests by finishing the current instruction, disabling interrupts, issuing INTA status and then automatically executing an RST 38h instruction (a one byte CALL to address 0038h). Again, the last service routine instruction is typically an RETI which resumes background program execution from the point of interruption.

Interrupt Mode 2 (IM2)

This mode is selected by executing an IM2 instruction. To use this mode, a table of interrupt service routine starting addresses (two bytes for each routine) must be maintained somewhere in IOP memory. The Z-80A responds to an interrupt request by finishing the current instruction, disabling interrupts, issuing INTA status and then reading an **interrupt vector byte** from the data bus. The Z-80A then forms an indirect CALL address by appending the interrupt vector byte (low order) to the contents of its I register (high order); the Z-80A then executes a CALL instruction to the indirect address so formed. Consequently, register I is typically loaded during system initialization with the service table page location (I-Page), and the interrupt vector then points to a specific table entry. The table entries are two byte values at adjacent memory locations N and N+1; N must be even valued (the interrupt vector LSB is assumed logic 0), the high order indirect address byte is located at N, and the low order address at N+1. Again, the last service routine instruction is typically a RETI.

Like NMI requests, maskable interrupt requests are serviced even when the Z-80A is in the HALTed state; but forcing C-Bus line $\overline{\text{WAIT}}$ active low prevents the current instruction from ending, and thus holds off interrupt servicing.

Of the three possible interrupt modes, IM2 would typically be chosen for its fast interrupt response and low programming complexity when several C-Bus interrupt sources are active. IM2 would definitely be chosen when using a Quadart as a C-Bus peripheral since all Quadart LSI devices may be programmed to supply IM2 interrupt vectors during INTA.

IM2 offers the additional advantage of freeing the one byte RST instructions for frequently used system CALLs. IM1 would typically be chosen when only a single C-Bus interrupt source is active. See Section 3.3 for a C-Bus interrupt programming example.

3.2 S-100 BUS INTERRUPTS

The IOP, along with other coresident S-100 bus boards, may issue maskable interrupt requests to the **host** CPU by forcing S-100 bus line $\overline{\text{INT}}$, [73], active low. Concurrently pending requests are prioritized by the Cromemco S-100 bus daisy chain. This topic is discussed later in this section. The IOP is not connected to S-100 bus line $\overline{\text{NMI}}$, [12], so it cannot issue NMI requests to the host CPU.

The IOP issues maskable interrupt requests to the host CPU as follows: (1) the IOP sets **Control Register** bit **Enable S-100 Interrupt Requests** -- this bit is latched and would typically be set during IOP initialization, and (2) the IOP outputs an interrupt vector byte to the IOP **Interrupt Vector** register. Writing the vector with bit **Enable S-100 Interrupt Requests** set automatically forces S-100 bus line $\overline{\text{INT}}$ active low; writing a vector with the bit reset is a null operation -- see Figure 26. Note that the IOP must output a vector byte to the **Interrupt Vector** register each time it issues a maskable interrupt request to the host CPU.

If host maskable interrupts are enabled, the host CPU acknowledges the request by forcing S-100 bus line sINTA, [96], active high. In response, IOP circuitry places the **Interrupt Vector** register contents on the S-100 data bus if no higher priority requests are pending. The IOP would typically write opcode for one of eight RST instructions to register **Interrupt Vector** if the host is operating in IM0, or it would write an indirect CALL table address if the host is operating in IM2 (if the host is operating in IM1, the value written is irrelevant since the data bus is ignored during INTA).

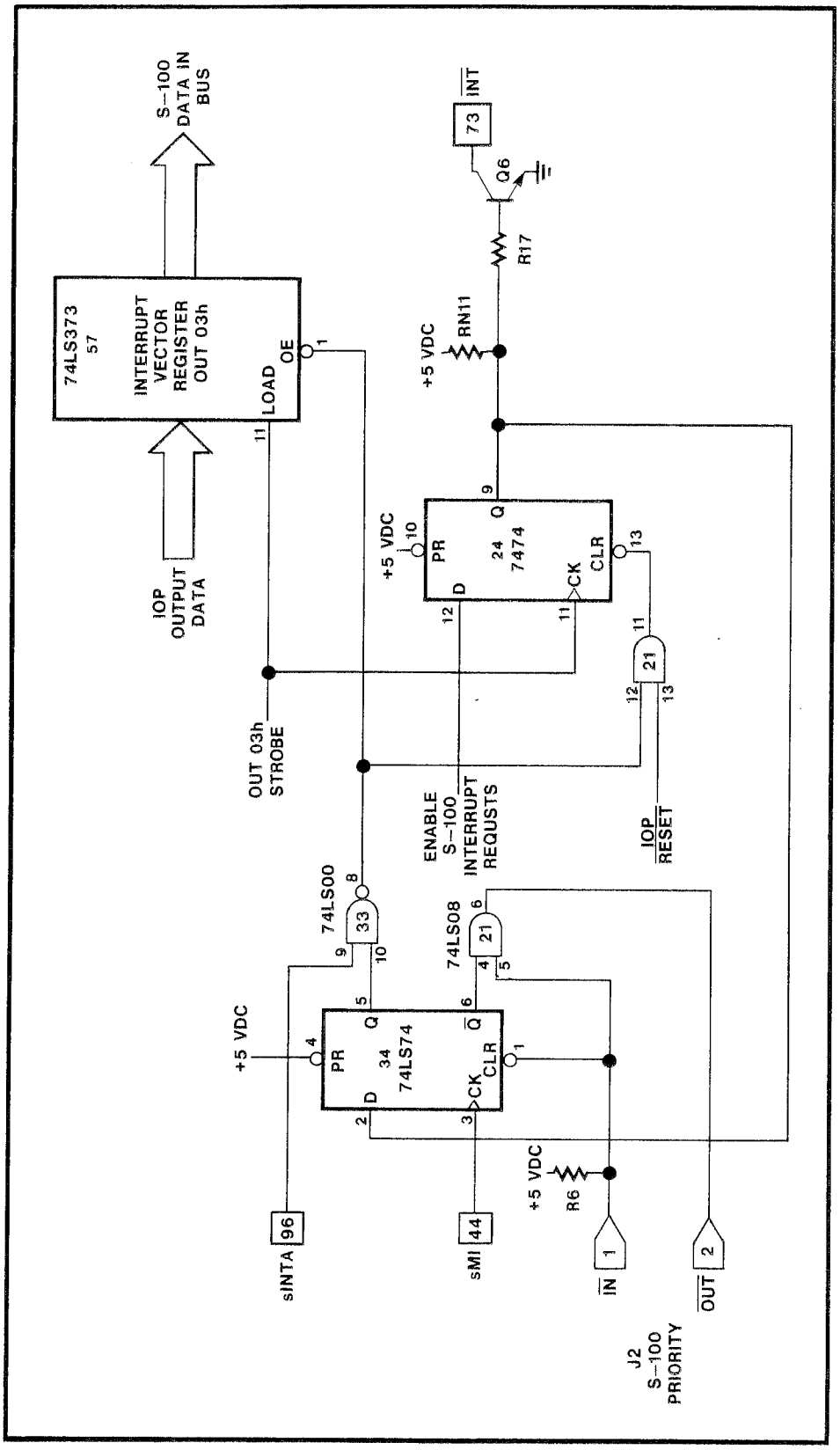


Figure 26: IOP S-100 INTERRUPT CIRCUITRY

The IOP together with the C-Bus peripherals which it controls may be thought of as an S-100 bus **task** which preprocesses data for the host CPU. For example, an IOP with one or more Quadarts function as a **serial I/O task** which manages synchronous and asynchronous data protocols, timing, level shifting, formatting, buffering, and error checking for the host. When two or more S-100 tasks issue maskable interrupt requests to the host CPU, then their requests are coordinated by Cromemco's S-100 Interrupt Priority daisy chain. The S-100 interrupt daisy chain is controlled by $\overline{\text{PRIORITY IN}}$ and $\overline{\text{PRIORITY OUT}}$ lines which are provided on the following Cromemco products: the **IOP**; the **PRI** line printer interface; the **TU-ART** serial interface; the **4FDC** and **16FDC** floppy disk interfaces; and the **WDI** Winchester hard disk interface. Each of these products also manages a system task for the host CPU, and a representative S-100 interrupt daisy chain task structure among them is shown in Figure 27.

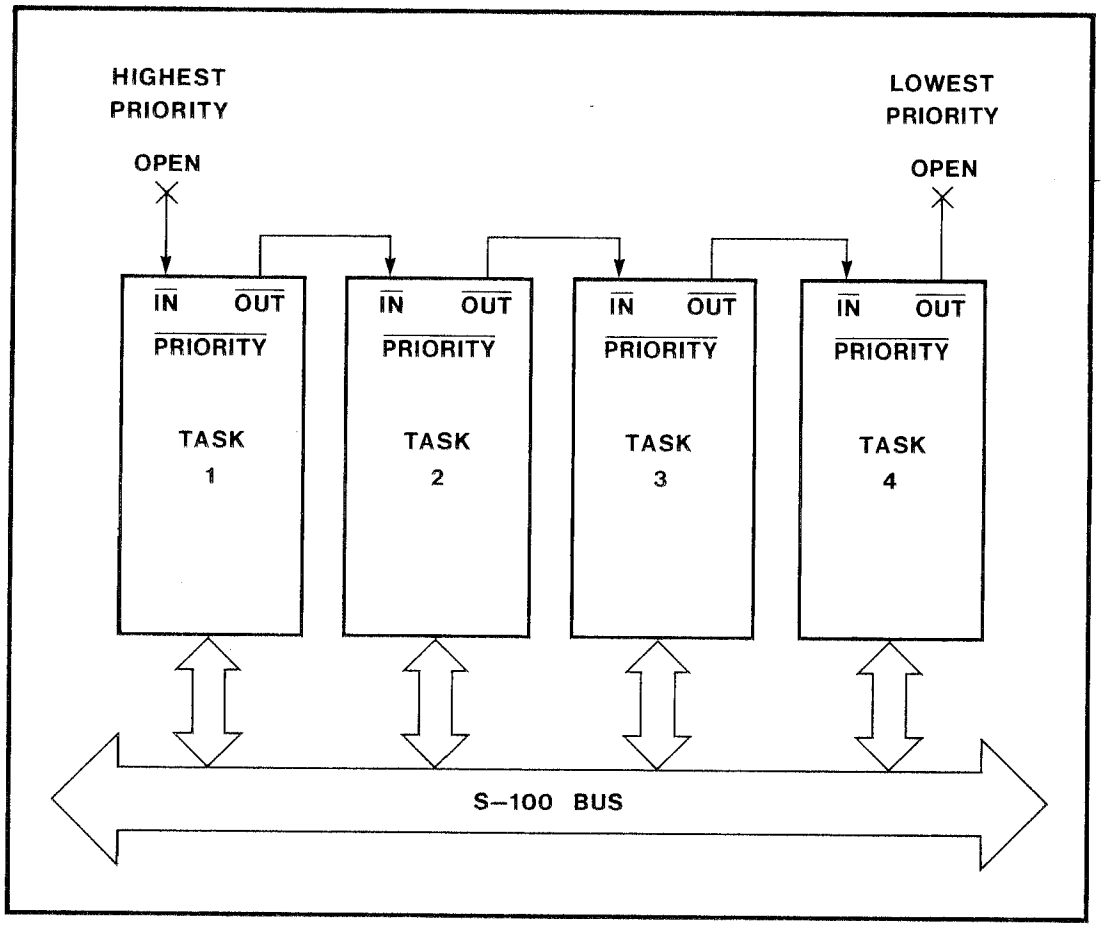


Figure 27: S-100 INTERRUPT DAISY CHAIN WIRING

The IOP PRIORITY IN and OUT lines are brought out at IOP connector J2. These lines behave as follows (again refer to Figure 26):

1. All S-100 bus tasks are wire-ANDed to S-100 bus line INT ; any task may assert INT active low at any time.

2. When a task asserts INT active low, then the PRIORITY OUT line of that task is forced low at the beginning of the next host CPU M1 (opcode fetch) machine cycle, unless it is already low due to a low PRIORITY IN signal from a higher priority task (see 3. below).

3. When task line PRIORITY OUT is forced low, this causes the PRIORITY OUT lines of all lower priority tasks to also go low. The low level is an inhibit signal which quickly ripples down to all lower priority tasks.

4. When the host acknowledges the maskable interrupt request by forcing S-100 bus line sINTA active high, only the task with a high PRIORITY IN line (the highest priority task requesting host service) may respond by: (1) placing its interrupt vector on the S-100 Data In bus, and (2) releasing line INT (lower priority tasks may continue to hold INT active low, however). Releasing line INT also causes the task's PRIORITY OUT line to go high at the beginning of the next host CPU M1 machine cycle, and this inactive high level propagates down the daisy chain at the rate of one task per M1 machine cycle until it reaches another task which is requesting host interrupt servicing.

5. Subsequent maskable interrupt requests proceed from step 1., and subsequent host acknowledgments proceed from step 4. after the host CPU has enabled maskable interrupt requests.

Note from the above that the S-100 bus PRIORITY IN and OUT lines behave differently than C-Bus PRIORITY IN and OUT lines. The S-100 bus lines resolve only **concurrently pending** interrupt requests in favor of the highest priority task; after servicing of that task has begun, any subsequent interrupt request is acknowledged as soon as maskable interrupts are enabled by the host, regardless of its relative interrupt priority (in such cases, the S-100 daisy chain again directs servicing to the highest priority request of those concurrently pending). In contrast, lower priority C-Bus requests are inhibited from interrupting any higher priority

service routine in progress until an **RETI** instruction is executed. Consequently, S-100 bus task interrupt priorities must be managed by the host software in concert with the S-100 daisy chain (for example, the highest priority task service routine might not execute an **EI** instruction until just before returning to the background program, whereas the lowest priority task service routine might execute an **EI** instruction immediately after entry).

3.3 INTERRUPT PROGRAMMING EXAMPLES

The following two program segments, one running in host memory and the other in IOP memory, illustrate host to IOP, and IOP to host, interrupts. In overview, the host program monitors the host console for a keystroke, and passes the resulting character to the IOP. After about .5 seconds the IOP program passes the character back to the host, which then displays it on the system console. In short, the two programs work in unison to simulate a host console read with a .5 second delayed echo.

HOST Program Description

The host program assumes a CDOS host system, as CDOS system calls are used to read from, and write to, the host console (see Cromemco's CDOS Operating System Instruction Manual, part number 023-0036). The host Z-80A is programmed to operate in IM0, and RST address 0010h is used for IOP interrupt servicing (0000h is used for CDOS reentry, 0008h is used to differentiate between CDOS and the Cromix CDOS simulator, 0030h is used for breakpoints in program **DEBUG**, and 0038h is used for **Invalid Jump** messages). The CDOS user program area starts at 0100h, and since Cromemco's linker/loader program **LINK** cannot load object code from 0000h to 00FFh (this area is reserved for CDOS), the RST 10h service routine object code is moved from its assembled location, 011Bh - 0126h, down to 0010h - 001Bh by the host program itself using the **LDIR** block move instruction.

```
CROMEMCO Z80 Macro Assembler version 03.07

                        0001
                        0002  ;       HOST PROGRAM 6
                        0003
                        0004  ;  Interrupt Programming Example.
                        0005  ;  See Text For Program Description.
                        0006
     (00CE)             0007  IBASE   EQU     0CEh            ;IOP Base Address.
     (00CF)             0008  OUTDATA EQU     IBASE+01h       ;Output Data Register.
     (00CF)             0009  INDATA  EQU     IBASE+01h       ;Input Data Register.
     (0005)             0010  CDOS    EQU     0005h           ;CDOS Call Address.
     (0002)             0011  WRTCON  EQU     2               ;Write Console CDOS call.
     (0080)             0012  RDCON   EQU     80h             ;Read Console (with no
                        0013                                  ;echo) CDOS call.
                        0014
                        0015          ;=======================
                        0016          ; Host Initialization and
                        0017          ; Background Program
                        0018          ;=======================
                        0019
     (0100)             0020          ORG     100h            ;Start of CDOS user area.
                        0021
0100 310010             0022  HOST6:  LD      SP,1000h        ;Locate stack.
                        0023
0103 211B01             0024          LD      HL,RST10        ;Relocate RST10 code from
0106 111000             0025          LD      DE,0010h        ;present memory location down
0109 010C00             0026          LD      BC,LAST-RST10   ;to proper location spanning
010C EDB0               0027          LDIR                    ;low RAM memory 0010h - 001Bh.
                        0028
010E ED46               0029          IM      0               ;Select IM0 mode (emphasis only).
0110 FB                 0030          EI                      ;Enable host maskable interrupts.
                        0031
0111 0E80               0032          LD      C,RDCON         ;Set up CDOS console read call.
0113 CD0500             0033  BAKGND: CALL    CDOS            ;Get keyboard character, load in
0116 D3CF               0034          OUT     (INDATA),A      ;IOP Input Data port for reading
0118 76                 0035          HALT                    ;and halt while waiting for
                        0036                                  ;interrupt from IOP.
0119 18F8               0037          JR      BAKGND          ;After interrupt, go back and get
                        0038                                  ;another keyboard character.
                        0039
                        0040          ;=======================
                        0041          ; IOP Interrupt Service
                        0042          ;=======================
                        0043
                        0044          ; The following code is moved down to memory
                        0045          ; 0010h - 001Bh by the Block Move instruction
                        0046          ; in the initialization segment above.
                        0047
011B D9                 0048  RST10:  EXX                     ;Switch to Z80 primed registers.
                        0049
011C 0E02               0050          LD      C,WRTCON        ;Set up CDOS console write call.
011E DBCF               0051          IN      A,(OUTDATA)     ;Get character back from IOP and
0120 5F                 0052          LD      E,A             ;write to console for viewing.
0121 CD0500             0053          CALL    CDOS            ;
                        0054
0124 D9                 0055          EXX                     ;Restore unprimed Z80 registers,
0125 FB                 0056          EI                      ;enable host maskable interrupts
0126 C9                 0057          RET                     ;and return to background program.
                        0058
                        0059  LAST:                           ;End Of RST10 Code For Block Move.
                        0060
                        0061          ;=======================
                        0062
0127 (0100)             0063          END     HOST6
```

```
CROMEMCO Z80 Macro Assembler version 03.07

                    0001
                    0002  ;       IOP PROGRAM 6
                    0003
                    0004  ;  Interrupt Programming Example.
                    0005  ;  See Text From Program Description.
                    0006
      (0001)        0007  INDATA  EQU     01h             ;Input Data Register.
      (0001)        0008  OUTDATA EQU     01h             ;Output Data Register.
      (0002)        0009  CONTROL EQU     02h             ;Control Register.
      (0002)        0010  FLAGS   EQU     02h             ;Flag Register.
      (0003)        0011  INTVEC  EQU     03h             ;Interrupt Vector Register.
      (00D7)        0012  RST10   EQU     0D7h            ;RST 10h opcode vector.
      (0006)        0013  IDA     EQU     6               ;Input Data Available bit D6.
                    0014
      (4000)        0015          ORG     4000h           ;Start of IOP RAM memory.
                    0016
                    0017          ;=======================
                    0018          ; IOP Initialization and
                    0019          ; Background Program
                    0020          ;=======================
                    0021
4000  310060        0022  IOP6:   LD      SP,6000h        ;Locate stack.
                    0023
4003  3E05          0024          LD      A,00000101b     ;Set bits Enable IOP Interrupts
4005  D302          0025          OUT     (CONTROL),A     ;and Enable S-100 Interrupt Requests.
                    0026
4007  3E40          0027          LD      A,40h           ;Initialize I-Page to 40h,
4009  ED47          0028          LD      I,A             ;
400B  ED5E          0029          IM      2               ;select Interrupt Mode 2, and
400D  FB            0030          EI                      ;enable IOP maskable interrupts.
                    0031
400E  18FE          0032  BAKGND: JR      BAKGND          ;
                    0033
                    0034          ;=======================
                    0035          ; Host Interrupt Service
                    0036          ;=======================
                    0037
                    0038  HOSTSRV:                        ;Automatic 0FEh vector results
                    0039                                  ;in CALL to this routine.
                    0040
4010  F5            0041          PUSH    AF              ;Save used registers on stack.
4011  E5            0042          PUSH    HL              ;
                    0043
4012  DB02          0044          IN      A,(FLAGS)       ;Exit service routine if no
4014  CB77          0045          BIT     IDA,A           ;input data available (e.g.,
4016  2810          0046          JR      Z,EXIT          ;when host reads output data).
                    0047
4018  210000        0048          LD      HL,0            ;Pause for about 1/2 second,
401B  2B            0049  PAUSE:  DEC     HL              ;
401C  7C            0050          LD      A,H             ;
401D  B5            0051          OR      L               ;
401E  20FB          0052          JR      NZ,PAUSE        ;
                    0053
4020  DB01          0054          IN      A,(INDATA)      ;get character from host,
4022  D301          0055          OUT     (OUTDATA),A     ;latch in output port for host reading,
4024  3ED7          0056          LD      A,RST10         ;then interrupt host by writing
4026  D303          0057          OUT     (INTVEC),A      ;RST 10h opcode to Interrupt Vector port.
                    0058
4028  E1            0059  EXIT:   POP     HL              ;restore registers,
4029  F1            0060          POP     AF              ;
402A  FB            0061          EI                      ;enable IOP maskable interrupts
402B  ED4D          0062          RETI                    ;and return from interrupt.
                    0063
                    0064          ;=======================
                    0065          ; IM2 Indirect Jump Table
                    0066          ;=======================
                    0067
      (40FE)        0068          ORG     40FEh           ;Since I-Page = 40h and Interrupt
                    0069                                  ;Vector = 0FEh, the IOP's Z80A goes
40FE  1040          0070          DW      HOSTSRV         ;here for IM2 indirect CALL address.
                    0071
                    0072          ;=======================
                    0073
4100  (4000)        0074          END     IOP6
```

After locating the service routine, the host program selects IM0 (the default mode after a Z-80A reset), enables host maskable interrupts, and then reads a keyboard character. The **read console without echo** CDOS system call returns after a keystroke occurs with the ASCII character in register A. The program then writes the character to the IOP **Input Data** register (this event generates an interrupt on the IOP board) and halts while waiting for an IOP interrupt. Instead of looping back to the CDOS **console read**, the program halts to prevent two CDOS system calls from becoming nested during interrupt servicing, since CDOS system calls are non-reentrant.

The host program halts for about .5 seconds waiting for an interrupt from the IOP (the Z-80A internally executes **NOP**s while halted). The IOP places RST 10h opcode on the S-100 Data In lines during interrupt acknowledge, and this one byte call instruction vectors the host program to address 0010h for interrupt servicing. The interrupt service routine switches to Z-80A primed registers, reads the echoed character from IOP register **Output Data** (this event also generates an interrupt on the IOP board), then writes the character to the host console using the **console write** CDOS system call. The routine then restores Z-80A unprimed registers, enables host maskable interrupts, and returns to the host background program. The **RET** instruction pops the address of the instruction following the **HALT** into the Z-80A Program Counter, which here means that **JR BAKGND** is executed next. The process described above is then repeated starting with the **read console without echo** CDOS system call, and so on.

IOP Program Description

The IOP program spans 4000h - 402Bh, and the IM2 jump table spans 40FEh - 40FFh, in IOP RAM memory. The stack is placed at 6000h (5FFFh and downward) to avoid conflict with Cromemco program **IOPDEBUG** (6800h - 7FFFh), which would typically be coresident during program verification. The IOP program initializes the IOP interrupt circuitry by setting **Control Register** bits **Enable IOP Interrupts** (D2) and **Enable S-100 Interrupt Requests** (D0). Setting bit **Enable IOP Interrupts** causes an internal IOP interrupt each time the host writes to, or reads from, the IOP ports, and IOP on board circuitry automatically supplies interrupt vector FEh to itself during interrupt acknowledge. In this application, the IOP is interrupted when the host writes a keyboard character to IOP port **Input Data**, and also when the host reads the echoed character from **Output Data.** In both

cases interrupt service routine **HOSTSRV** is entered as a result.

Setting bit **Enable S-100 Interrupt Requests** causes the IOP to interrupt the host Z-80A each time a vector is written to IOP register **Interrupt Vector.** In this example, the IOP writes **RST 10h** opcode (D7h) to **Interrupt Vector** since the host Z-80A operates in IM0, and the IOP interrupt service routine begins at 0010h in host memory. The program selects I-Page 40h, selects IM2 and enables IOP maskable interrupts to complete IOP initialization. A background program, consisting of a **jump to itself** instruction, is then entered while waiting for interrupts from the host system.

When the host writes a character to register **Input Data,** an interrupt is generated, I-Page 40h and interrupt vector FEh are combined to form indirect call address 40FEh, and address 4010h found there vectors the Z-80A to routine **HOSTSRV.** This routine verifies that the interrupt occurred because **Input Data** is available (and **not** because the host read register **Output Data**), then it pauses .5 seconds, reads the character from **Input Data** and writes it to **Output Data,** generates a host interrupt by loading port **Interrupt Vector,** reenables IOP maskable interrupts, and returns from the interrupt. The host responds to its interrupt by reading register **Output Data,** which in turn interrupts the IOP. This time, routine **HOSTSRV** determines that **Input Data** is not available, so it merely reenables IOP maskable interrupts and returns.

3.4 C-BUS DMA

There are two Direct Memory Access categories which relate to the IOP board: **C-Bus DMA** and **S-100 Bus DMA.** C-Bus DMA refers to a state in which a C-Bus peripheral controls the C-Bus address, data and control lines for the purpose of exchanging data with another C-Bus peripheral without IOP intervention. In contrast, S-100 Bus DMA refers to a state in which an S-100 bus peripheral controls the S-100 bus address, data and control lines for the purpose of exchanging data with another S-100 bus peripheral (including the IOP) without host CPU intervention. All data exchanged with the IOP in this fashion must pass through the IOP's two I/O ports, since they constitute its only interface to the S-100 bus. This section discusses C-Bus DMA, while Section 3.5 covers S-100 bus DMA.

C-Bus DMA requests and acknowledgments are managed by IOP software, **not** by a direct hardware link to the IOP Z-80A (refer to IOP Schematic Diagram and note that Z-80A input pin $\overline{\text{BUSRQ}}$ is tied inactive high and output pin $\overline{\text{BUSACK}}$ is floating). A C-Bus peripheral makes a DMA request to the IOP by asserting line $\overline{\text{CPU DISCONNECT}}$, <40>, active low. This line is fed through an inverter to **Flag Register** bit **CPU Disconnect**, D4, which the IOP may then poll until set. When the IOP polls bit **CPU Disconnect** set, it may optionally acknowledge the request by setting **Control Register** bit **C-Bus Available**, D4, or it may ignore the request. If bit **C-Bus Available** is set, then the IOP relinquishes control of the C-Bus address, data and control lines, as shown in Figure 28, by disabling its 3-state drivers, and it also forces C-Bus line $\overline{\text{BUS AVAILABLE}}$, <41>, low as an acknowledgment to the peripheral.

While **Control Register** bit **C-Bus Available** is set and a C-Bus DMA transfer is in progress: (1) the program in IOP memory continues to run -- this is in sharp contrast to what would occur were the Z-80A to enter a true Bus Acknowledge state in response to a hardware Bus Request, (2) the running IOP program would typically poll **Flag Register** bit **CPU Disconnect** until reset by the peripheral to signal an end to the DMA transfer, (3) IOP on board memory and I/O ports are **not** accessible to the peripheral for DMA transfers, (4) the IOP continues to respond to signals on C-Bus lines $\overline{\text{NMI}}$, $\overline{\text{INT}}$, $\overline{\text{WAIT}}$, $\overline{\text{CPU DISCONNECT}}$, and $\overline{\text{PRI 3}}$, and it also continues to drive C-Bus lines $\overline{\text{HALT}}$, ϕ, and $\overline{\text{BUS AVAILABLE}}$, and (5) the IOP may continue normal communications, polled or interrupt driven, with the host system over the S-100 bus.

When the C-Bus DMA transfer is complete, the peripheral drives C-Bus line $\overline{\text{CPU DISCONNECT}}$ inactive high. The IOP detects this event by polling bit **CPU Disconnect** reset, and it would then reset bit **C-Bus Available** in response. This drives C-Bus line $\overline{\text{BUS AVAILABLE}}$ inactive high, and returns control of the C-Bus address, data and control lines to the IOP Z-80A.

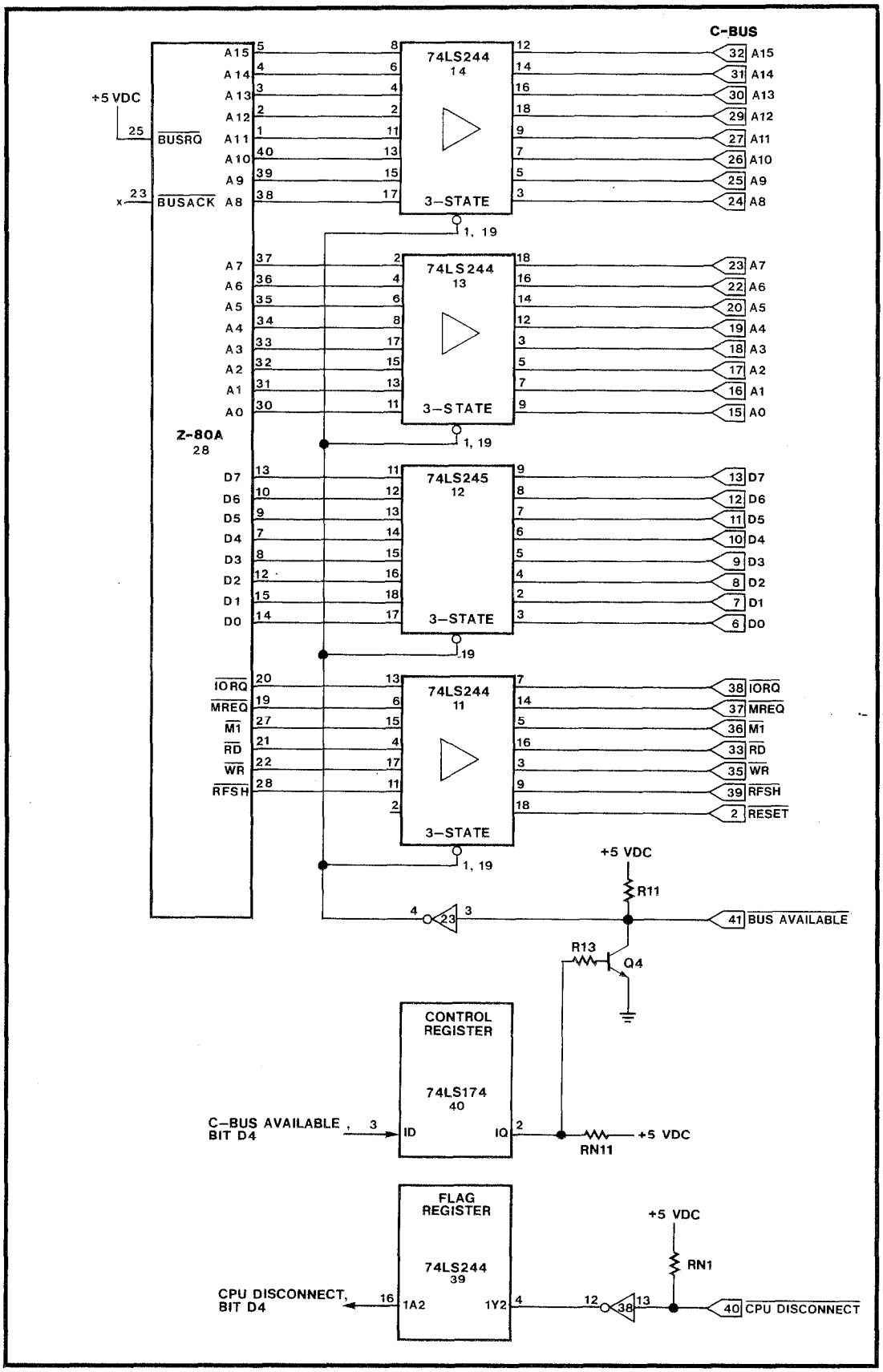


Figure 28: C-BUS DMA REQUEST AND ACKNOWLEDGE CIRCUITRY

3.5 S-100 BUS DMA

S-100 bus DMA requests and acknowledgments are under the supervision and control of the host CPU. The IOP cannot act as a DMA controller, nor can it issue DMA requests to the host CPU since S-100 bus line $\overline{\text{HOLD}}$, [74], is not connected to the IOP board. The IOP can participate in DMA transfers with other S-100 bus peripherals under certain conditions, however IOP board behavior during S-100 bus DMA cycles (line pHOLD, [26], active high) and during non-DMA cycles (pHOLD inactive low) is determined by the IOP **DMA/NORM** switch setting -- see Figure 29.

Before discussing the switch setting differences, note that regardless of switch position, and regardless of S-100 cycle type (DMA or non-DMA): (1) the program in IOP memory continues to run, (2) this running program continues to control IOP data, command and status transfers, C-Bus interrupts and IOP internal interrupts, (3) all data, commands and status transferred between S-100 devices and the IOP continue to pass through IOP registers **Output Data**, **Input Data**, **Status Register** and **Command Register**, and (4) since all data, command and status transfers with the IOP are under program control, the transfer rates may be limited by the IOP Z-80A program execution speed. That is, from the viewpoint of an S-100 device (like the host CPU or a DMA controller), the IOP **always** looks like four software controlled I/O ports. An S-100 device never has direct read/write access to internal IOP memory, nor to internal IOP registers -- they are always under control of the program running from IOP memory.

Switch Position = NORM

This is the typical IOP switch setting. Here, S-100 devices have read/write access to the four IOP I/O registers **only** during non-DMA, or **normal** host machine cycles (when pHOLD is inactive low). Each such read/write operation may be programmed to generate an IOP internal interrupt by setting **Control Register** bit **Enable IOP Interrupts**, as explained in Section 3.1.

With this switch setting, the four IOP I/O registers do **not** respond to S-100 read/write operations directed to their port addresses during S-100 DMA cycles (when pHOLD is active high), nor does the IOP internal interrupt circuitry respond to these read/write operations. Consequently, both IOP port addresses (normally CEh and CFh) may be used by other S-100 peripherals during S-100 DMA cycles only.

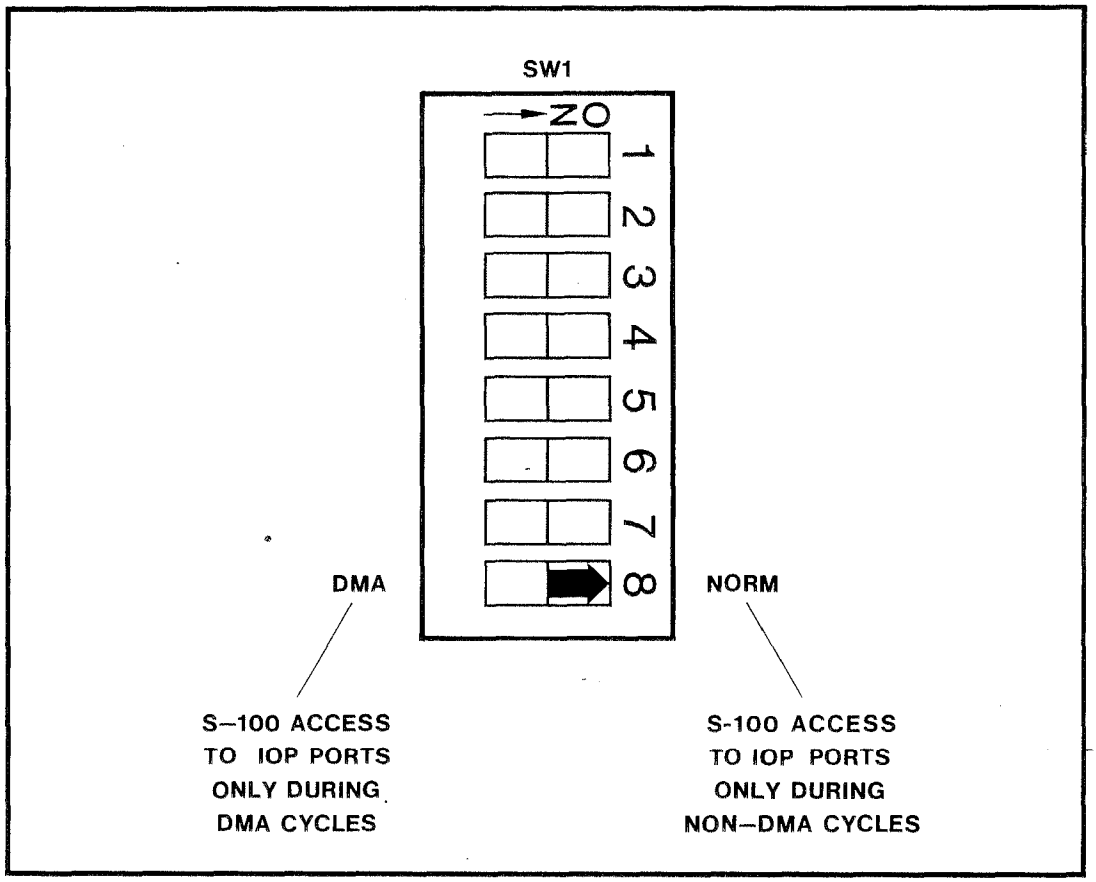


Figure 29: DMA/NORMAL SWITCH

Switch Position = DMA

This unusual switch setting would make the IOP operate exclusively as an S-100 DMA peripheral. Here, S-100 devices have read/write access to the four IOP I/O registers **only** during S-100 DMA cycles (when pHOLD is active high). Each such read/write operation may be programmed to generate an IOP internal interrupt by setting **Control Register** bit **Enable IOP Interrupts**, as explained in Section 3.1.

With this switch setting, the four IOP I/O registers do **not** respond to S-100 read/write operations directed to their port addresses during S-100 **normal** host machine cycles (when pHOLD is inactive low), nor does the IOP internal interrupt circuitry respond to these read/write operations. Consequently, both IOP port addresses may be used by other S-100 peripherals during S-100 **normal** cycles only.

Chapter 4

IOP THEORY OF OPERATION

The major internal IOP subsystems are discussed in the following sections. The last section of this chapter presents the C-Bus waveforms for each possible IOP processor cycle type. Refer to the IOP Schematic Diagram to cross reference device numbers which are called out in the text.

4.1 IOP POWER

The IOP draws its power from the S-100 bus. Bus lines +8 VDC are regulated to +5 VDC @ 1.75 Amps (max) by IC31 and IC48. Bus lines +18 VDC and -18 VDC are regulated to +12 VDC @ 100 mAmps and -5 VDC @ 25 mAmps by IC49 and Zener diode D1, respectively.

4.2 CLOCKS

An 8.000 MHz crystal controls VCO IC63 which drives a divide-by-two flip flop to generate a 4.000 MHz, 50% duty cycle square wave at IC3 pin 6. This signal is inverted at IC1 pin 12, and conditioned by R30, R31, C64 and Q8 to drive Z-80A input ϕ to meet the Z-80A Clock Input Voltage High specification. Signal IC3 pin 6 also drives C-Bus signal $\overline{\phi}$ through driver IC5. Note that C-Bus signal $\overline{\phi}$ is inverted with respect to IOP Z-80A clock input ϕ.

4.3 THE IOP CPU

The IOP central processing unit is a Z-80A clocked at 4.000 MHz. The Z-80A address lines drive the internal IOP address bus through buffer/drivers IC15 and IC16, C-Bus lines A0 - A15 through buffer/drivers IC13 and IC14. The Z-80A data lines exchange data with the internal IOP data bus over bus transceiver IC42, and with C-Bus lines D0 - D7 over transceiver IC12. The Z-80A status lines drive the internal IOP status bus through buffer/drivers IC27, and the C-Bus status lines through buffer drivers IC11. The C-Bus buffer/drivers and transceivers function both to meet C-Bus drive requirements, and to isolate the IOP from the C-Bus while being managed by another controller during C-Bus DMA cycles.

Z-80A input line $\overline{\text{BUSRQ}}$ is tied inactive high, and its $\overline{\text{BUSACK}}$ output pin is unconnected since C-Bus DMA is under IOP software control. Z-80A control lines $\overline{\text{NMI}}$, $\overline{\text{INT}}$ and $\overline{\text{WAIT}}$ may be pulled active low at any time by C-Bus devices wire-ANDed to them. Additionally, lines $\overline{\text{INT}}$ and $\overline{\text{WAIT}}$ are driven by internal IOP circuitry. Line $\overline{\text{INT}}$ is pulled active low (Q1 turned on) at the beginning of an M1 machine cycle if an internal IOP interrupt is pending (IC32 pin 9 set) and line $\overline{\text{PRI 3}}$ is inactive high at IC2 pin 5. Line $\overline{\text{WAIT}}$: (1) is never pulled active low by on board IOP circuitry if IC4 pin 1 is tied to ground potential as described above in Section 2.4, or (2) is pulled active low for one 250 nSec Wait State during each IOP ROM memory read cycle if IC4 pin 1 is pulled up to +5 VDC by leaving jumper option **M2** open circuited, or (3) is pulled active low for one 250 nSec Wait State during each IOP Z-80A M1 (fetch) cycle if jumper **M2** is installed. The $\overline{\text{WAIT}}$ line is not asserted active low during any other type of IOP machine cycle.

Z-80A line $\overline{\text{RESET}}$ is held low for approximately 150 mSec after +8 VDC is applied to the S-100 bus by R4, C1, Q2 and associated circuitry. $\overline{\text{RESET}}$ is also forced active low when **Control Register** bit **S-100 Reset** is set under IOP software control (IC40 pin 15 goes high which turns on Q7); the line is automatically released shortly afterward when bit **S-100 Reset** is hardware reset by feedback path IC21 pin 3 to IC40 pin 1.

4.4 IOP DYNAMIC RAM

The IOP is shipped with 16 Kbytes of dynamic RAM. The memory chip used is a 150 nSec version of the 4116. No Wait States are inserted when either reading from, or writing to, IOP 4116 RAM. Memory mapping PROM IC10 locates the 16 Kbytes of RAM in the IOP memory map (see Section 2.5), and it also generates select signal $\overline{\text{RAM}}$ (IC10 pin 13) which is used to generate 4116 read/write enable signal $\overline{\text{CAS}}$ (4116 pin 15). The factory shipped memory mapping PROM locates 4116 RAM at 4000h - 7FFFh in the IOP memory map.

The 4116 is a dynamic RAM with seven multiplexed address input lines, A0 - A6. When IOP 4116 is either written to, or read from by the IOP Z-80A, multiplexers IC29 and IC30 first present row address A0-A6 to the 4116, and the 4116 then internally latches these bits on the falling edge of $\overline{\text{MREQ}}$ during T1 (latching signal $\overline{\text{RAS}}$ goes low at this time). The multiplexers then present column address A7 - A13 to the 4116 on the rising edge

of ϕ at start of T2. Delay network R23 - C63 causes $\overline{CAS}$ to go low soon after address A7 - A13 appears. The falling edge of $\overline{CAS}$ latches the seven high order address bits, and it also serves as a 4116 read/write enable.

IOP 4116 dynamic RAM is refreshed by the Z-80A during M1 (instruction fetch) cycles. The Z-80A places the contents of its R (refresh) register on address lines A0 - A6 during each M1 cycle while status lines $\overline{RFSH}$ and $\overline{MREQ}$ are both active low; $\overline{RAS}$ goes low during this interval to produce a RAS-only refresh cycle. The R register value is incremented between M1 cycles, so it cycles through a complete seven bit binary count with each set of 128 M1 cycles executed. The 4116 must be completely refreshed (a RAS cycle of any type to each row address) every 2 mSec, consequently **128 instruction fetch cycles must be executed during every 2 mSec interval.** Put another way, the IOP Z-80A must not be inhibited from executing at least 128 instructions during every 2 mSec interval, otherwise 4116 RAM memory may lose its data.

There are only two events (other than a loss of power) which can inhibit the Z-80A from executing instructions: (1) an external device drives Z-80A pin $\overline{BUSRQ}$ low which forces the Z-80A to enter a HOLD (DMA) state, or (2) an external device drives Z-80A pin $\overline{WAIT}$ low which forces the Z-80A to enter a WAIT state (note that the Z-80A internally executes NOPs while HALTed, and thus continues to supply refresh in this condition). The $\overline{BUSRQ}$ pin of the IOP Z-80A is strapped high and C-Bus DMA is handled by the IOP software specifically to avoid a loss of refresh due to event (1). This leaves only event (2) as a potential refresh problem.

The 128 instruction executions per 2 mSec interval specification may be met by incorporating a hardware or software managed timeout interval on all C-Bus hardware designs capable of holding the $\overline{WAIT}$ line active low for extended time intervals, or by using memory and I/O devices requiring none, or only a few Wait States per access. Conservative estimates of the maximum number of Wait States which can be inserted by the slowest C-Bus memory, or by the slowest C-Bus I/O device on all accesses while still meeting the refresh specification might proceed as follows: for slow external C-Bus memory, assume that the Z-80A repeatedly fetches and executes its lengthiest instruction from this slow external memory -- 23 T-cycles and 6 M-cycles (there are several Z-80A instructions in this group). The IOP

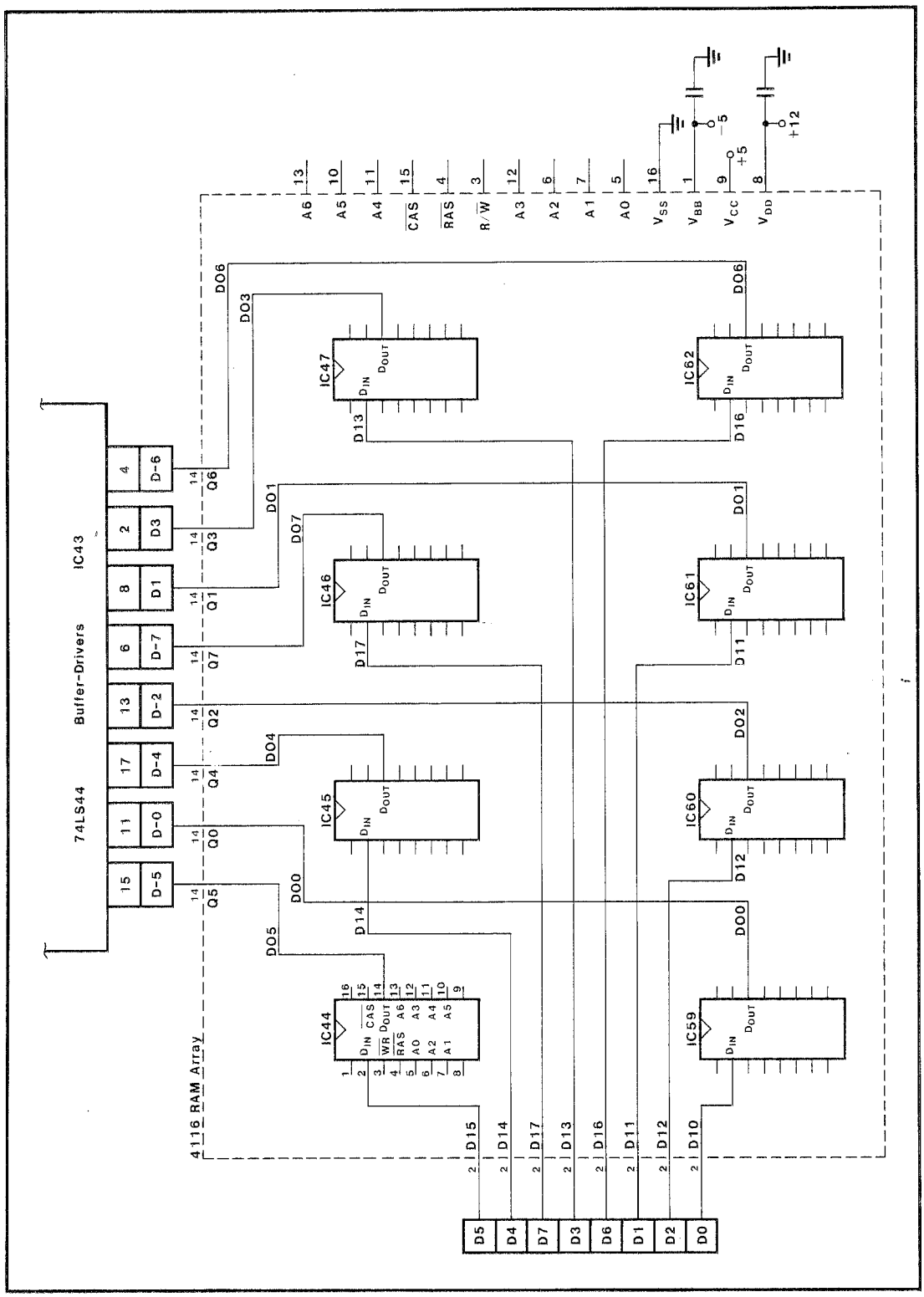


Figure 30: IOP 16K DYNAMIC RAM ARRAY

clock frequency is 4.000 MHz, so each T-cycle is 250 nSec long. Thus 5.75 uSec is required to execute one of these instructions with no Wait States. Executing 128 instructions every 2 mSec averages to 15.625 uSec per instruction, and this leaves 15.625 uSec - 5.75 uSec = 9.875 uSec available for Wait States per instruction. Since memory can be accessed on each instruction M-cycle (as with an **EX (SP),IX** instruction), 9.875 uSec/6 = 1.65 uSec or **6 Wait States** maximum may be inserted per memory cycle.

For slow external C-Bus I/O, assume that an **OUTI** instruction is repeatedly executed (the same analysis would apply to instructions **OUTD, INI** or **IND**). This I/O instruction is chosen for a worst case analysis rather than a simple **OUT**, or a block move **OTIR**, since it (or any of its three equivalents) requires the most time to transpire between Z-80A refresh cycles. The **OUTI** instruction requires 16 T-Cycles, or 4.000 uSec, to execute with no additional Wait States -- a fetch cycle followed by a memory read cycle to get the two byte opcode, another memory read cycle to get the data byte from memory, and an output cycle (which includes one automatically inserted Wait State) to write the data to an I/O port. With an average instruction execution time of 15.625 uSec to satisfy the 4116 refresh timing requirement, the average time allowed for additional Wait State insertions is then 15.625 uSec - 4.00 uSec = 11.625 uSec, or **46 Wait States** per **OUTI** instruction. The 46 Wait States may be distributed in any way among the three memory cycles and one output cycle. If, for example, the **OUTI** instructions are being executed from fast (no Wait State) memory, then all 46 Wait States could be allotted to the instruction output cycle. If the slow memory from the previous analysis is assumed, then 18 Wait States would be used by the three memory cycles, and 26 Wait States would be left over for the output cycle.

These estimates are quite conservative (worst case) and restrictive (the Wait States are assumed to be uniformly distributed over time rather than grouped in bursts). They are provided only to supply a refresh analysis starting point when designing custom built C-Bus peripherals which use C-Bus line $\overline{\text{WAIT}}$ to synchronize slow data transfers with the IOP. It should be noted that a refresh analysis is unnecessary when using a stand-alone IOP in a host system, or an IOP with Cromemco designed C-Bus peripherals, since these products meet all 4116 dynamic RAM refresh requirements.

4.5 IOP ROM MEMORY

The IOP features four 24-pin ROM sockets which may be filled with a mix of device types, or left vacant. Memory mapping PROM IC10 defines the IOP memory map, and it also generates the ROM select strobes (IC10 pin 12 generates $\overline{CS0}$, the strobe for socket ROM0, and so on -- see Section 2.5). The factory shipped memory mapping PROM configures all four ROM sockets for TI 2516 or Intel 2716 (or equivalent) devices; ROM0 spans addresses 0000h through 07FFh (2 Kbytes), ... , ROM3 spans 1800h through 1FFFh.

The IOP user may wish to program one or more custom 74191 ROMs to accommodate different mixes of the following device types, or their generic equivalents:

EPROMs:	Intel 2716, 2732
	Texas Instruments 2516, 2532
Masked ROMs:	Texas Instruments 4732, 4764

Reference to Figure 3 and the manufacturer's data sheets shows that ROM pin 20 may be used as an active low read strobe for all device types. The use of ROM pins 18 and 21 differ from device to device, but all variations may be managed by properly programming the memory mapping PROM. Note that the IOP **cannot** program EPROMs. In IOP applications requiring 8 Kbytes or less of ROM code, the IOP user should consider employing a Cromemco **32K BYTESAVER** board to efficiently serve a dual purpose: (1) programming TI 2516 and Intel 2716 EPROMs for host system and IOP use, and (2) providing a 32 Kbyte EPROM memory board for the host system. This approach provides the fastest concept-to-firmware path since modifications to the memory mapping PROM are not required. The **32K BYTESAVER** also provides a convenient means to tailor fit the **IOPMON** program code in 2516 EPROM to meet specific system requirements.

4.6 IOP INPUT/OUTPUT REGISTERS

Seven on board I/O registers function to: (1) exchange data, commands and status between the IOP and the host system, (2) supply interrupt vectors to the host processor, (3) control selected S-100 and C-Bus interrupt, DMA and reset functions, and (4) supply

selected interrupt, DMA, reset and handshake flags to the IOP Z-80A.

All seven I/O port bits are directly accessible to the IOP Z-80A, except for OUT 00h (**Status Register**) bits D7, D6 and D1. The outputs of data selector IC25 generate all on board input/output strobes. IC25 is enabled when input condition

(M1=0) AND (A7=A6=A5=A3=A2=0) AND (IORQ=1)

is true. In this case, inputs A1, A0 and $\overline{WR}$ to IC25 then select one-of-seven active low outputs, and the single active strobe then selects among ports OUT 00h - OUT 03h, or IN 00h - IN 02h. Output strobes latch internal data bus bits into targetted ports, while input strobes gate the data from targetted ports onto the internal data bus. For example, IC25 pin 9 is the output strobe to OUT 03h (**Interrupt Vector**) which latches the internal data bus contents into IC57 with a momentary high level at pin 11. IC25 pin 10 is the input strobe to IN 02h (**Flag Register**) which gates the inputs to 3-state octal driver IC39 onto the internal data bus with an active low level applied to pins 1 and 19 in unison.

Reference to the IOP Schematic Diagram reveals the following structure:

1. OUT 00h (**Status Register**) is formed by hex latch IC41, 3-state driver IC56, and associated circuitry. Note that OUT 00h bits D7, D6 and D1 are defined by IOP hardware, and not by the data bits output by the Z-80A.

2. OUT 01h (**Output Data**) is formed by octal latch IC58 and associated circuitry.

3. OUT 02h (**Control Register**) is formed by hex latch IC40. **C-Bus Clear**, bit D5, is sampled at IC36 pin 9 by strobe IC36 pin 10. Note that OUT 02h bits D7 and D6 are not used.

4. OUT 03h (**Interrupt Vector**) is formed by octal latch IC57 and associated circuitry.

5. IN 00h (**Command Register**) is formed by octal latch IC54 and associated circuitry.

6. IN 01h (**Input Data**) is formed by octal latch IC55 and associated circuitry.

7. IN 02h (**Flag Register**) is formed by 3-state driver IC39 and associated circuitry.

4.7 EXTERNAL C-BUS MEMORY AND I/O PORTS

The C-Bus cable supplies all Z-80A address, data and control lines (except $\overline{\text{BUSRQ}}$ and $\overline{\text{BUSAK}}$) in parallel to each C-Bus peripheral. Thus the IOP can support external C-Bus ROM or RAM memory (static or dynamic), memory mapped I/O ports, and I/O mapped ports. The IOP differentiates between **internal** and **external** address references with signal **INTERNAL** generated at IC4 pin 8. The Boolean expression for this signal is:

```
INTERNAL =  (IN or OUT cycle to ports 00h - 03h)
     OR     (RAM from memory mapping PROM active low)
     OR     (ROM from memory mapping PROM active low)
     OR     (INTA cycle to internal IOP interrupt).
```

In other words, INTERNAL is active high while the IOP Z-80A is exchanging data with its own internal registers and memory, or while it is reading a FEh vector while acknowledging an internal IOP interrupt (from host I/O with IOP registers). Figure 31 shows how signal INTERNAL selects between the internal IOP data bus and external C-Bus data lines D0-D7, and also how it control the direction of data through C-Bus transceiver IC12.

Refer to the figure and notice that transceiver IC12 is always enabled except during C-Bus DMA cycles when $\overline{\text{BUS AVAILABLE}}$ is active low. C-Bus data lines D0 - D7 are gated to the Z-80A data bus when IC12 pin 1 is high, and this occurs only when a C-Bus read cycle ($\overline{\text{RD}}$ = 0) is in progress, or when (INTERNAL = 0) AND ([RD = 1] OR [INTA = 1])is true. These two facts together imply that the C-Bus data lines D0 - D7 are actively driven, and mimic the Z-80A data bus, for all but C-Bus read cycles, or during C-Bus DMA.

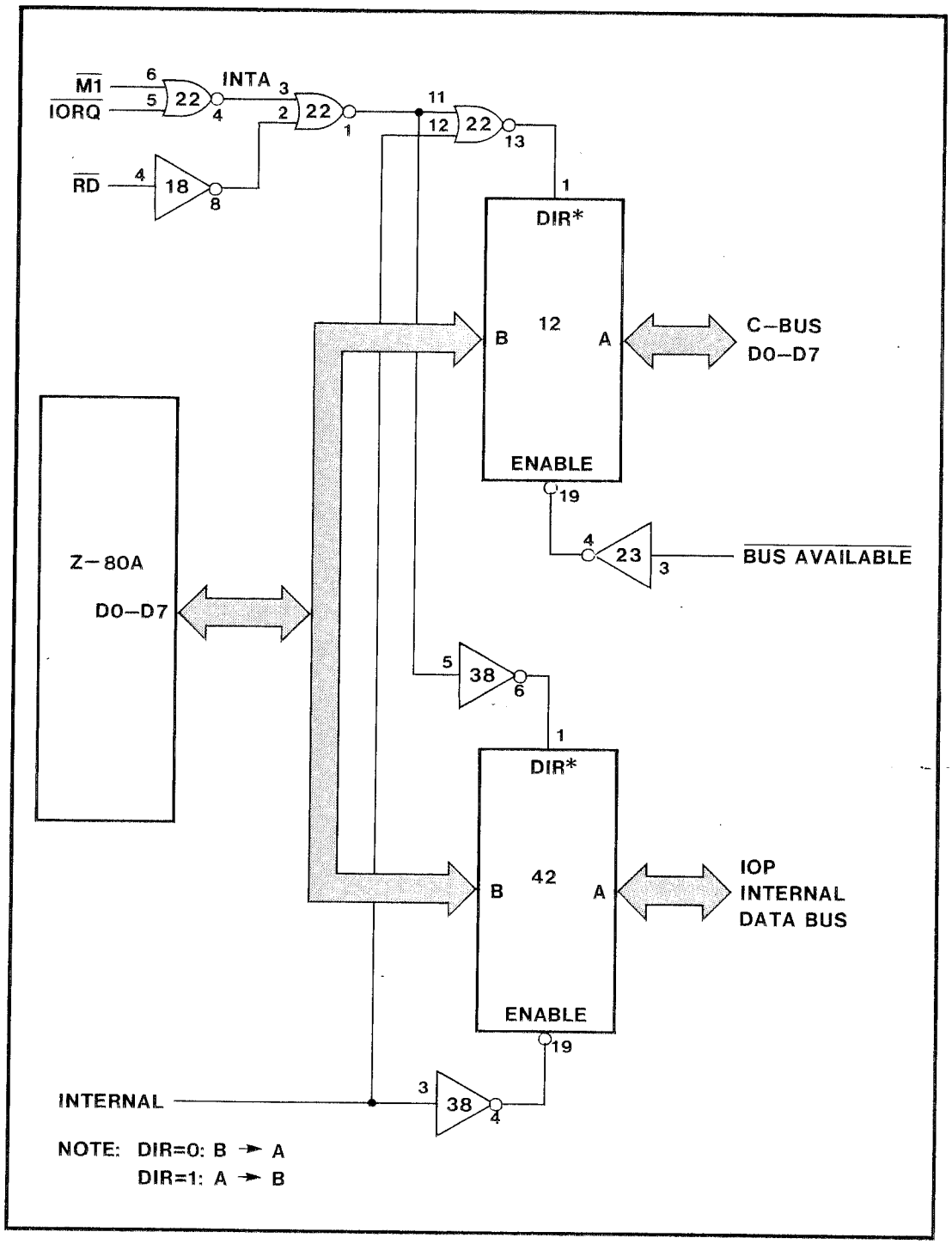


Figure 31: IOP INTERNAL/EXTERNAL DATA BUS CONTROL

This, together with further examination of Figure 31 and the IOP Schematic Diagram, implies the following facts:

1. The IOP Z-80A can write data to either its on board RAM memory or to external C-Bus RAM memory. In cases where the addresses of internal and external memory overlap, data is written in **parallel** to both destinations. Memory writes to on board ROM sockets should be avoided to preclude internal data bus conflicts.

2. The IOP Z-80A can output data to either its on board registers OUT 00h - OUT 03h, or to external C-Bus output ports. In cases where the internal and external port addresses overlap, data is output in **parallel** to both destinations.

3. The IOP Z-80A can read data from either its on board memory (RAM or ROM), or it can read from external C-Bus memory (ROM or RAM). In cases where the addresses of internal and external memory overlap, only on board memory is read -- **not** data from external memory. While such memory read cycles are in progress, the C-Bus data lines are actively driven by transceiver IC12, and are made to mimic the Z-80A data lines.

4. The IOP Z-80A can input data from either its on board registers IN 00h - IN 02h, or from external C-Bus input ports. In cases where the internal and external port addresses overlap, only data from on board registers is read -- **not** data from external input ports. While such input cycles are in progress, the C-Bus data lines are actively driven by transceiver IC12, and are made to mimic the IOP Z-80A data lines.

5. Typically, any external C-Bus memory or I/O ports would be assigned addresses which are disjoint with those of internal IOP register and memory. This means that, assuming the standard IOP memory map, external memory would occupy any addresses in the range 2000h-3FFFh or 8000h-FFFFh. Likewise, external output ports would be assigned port addresses in the range OUT 04h - OUT FFh, and external input ports would be assigned port addresses in the range IN 03h - IN FFh.

6. External C-Bus peripherals are responsible for controlling C-Bus line WAIT* during slow memory or I/O data transfers. The IOP can only insert Wait States when its internal ROM memory is read.

7. External C-Bus dynamic RAM memory is responsible for providing its own refresh. Note that the IOP Z-80A may be used to supply transparent M1 refresh cycles to external C-Bus dynamic RAM memory provided: (1) C-Bus line $\overline{\text{WAIT}}$ is properly managed (see Section 4.5), and (2) C-Bus DMA cycles do not interfere with refresh by floating the C-Bus address and control lines for extended intervals.

4.8 S-100 BUS INTERFACE

The IOP is interfaced to the S-100 bus through two input ports, two output ports, interrupt request/acknowledge circuitry, and S-100 bus reset circuitry.

S-100 address lines A7 - A0 are compared to the switch defined IOP base address (Ibase) by comparators IC52 and IC53. Line pHOLD is also compared to switch DMA/NORM, and if both the address and cycle type (DMA or non-DMA) agree with the switch settings, line IC53 pin 6 goes high to enable port strobes from decoder IC51. S-100 input cycles are signaled by status pDBIN = logic 1 and sINP = logic 1; output cycles are signaled by status sOUT = logic 1 and $\overline{\text{pWR}}$ = logic 0.

Active low input port strobes from IC51 enable the 3-state outputs of bus driver IC56 (**Status Register**) and octal latch IC58 (**Output Data**); these enabled outputs in turn drive the S-100 Data In bus lines. Active low output port strobes from IC51 gate data from the S-100 Data Out bus lines into octal latches IC54 (**Command Register**) and IC55 (**Input Data** -- this strobe is inhibited by flip flop IC33/IC20 until the previous data is read, or until an internal IOP reset occurs).

An active low on any of the four IC51 strobe lines causes IC50 pin 8 to go high which sets flip flop IC32 pin 9, provided **Control Register** bit **Enable IOP Interrupts** is set at IC32 pin 12. This event signals an internal IOP interrupt request (from host I/O with IOP registers), and the request is passed on to Z-80A pin $\overline{\text{INT}}$ through priority logic IC2 and associated circuitry.

The IOP issues S-100 maskable interrupt requests to the host processor by turning on transistor Q6 which is wire-ANDed to S-100 bus line $\overline{\text{INT}}$. The host processor acknowledges the request with status sINTA = logic 1. If line S-100 $\overline{\text{PRIORITY IN}}$ is inactive high at IC34 pin 1, the interrupt acknowledge cycle gates the contents of octal latch IC57 (**Interrupt Vector**) onto the S-100 Data

In bus lines as the INTA fetch cycle begins (when status line sM1 goes high).

The IOP resets the S-100 bus host system by turning on transistor Q7 which is wire-ANDed to S-100 bus line $\overline{\text{RESET}}$. The transistor is turned on by setting **Control Register** bit **S-100 Reset** which forces IC40 pin 15 high. A low at the collector of Q7 is fed back to the base of Q2 to simulate a POC, and this event resets the IOP Z-80A and also drives IC21 pin 3 low. This signal in turn resets all IOP internal circuitry including **Control Register** bit **S-100 Reset**, which finally releases S-100 bus line $\overline{\text{RESET}}$ by turning Q7 off.

4.9 IOP INTERRUPT LOGIC

There are two general categories of interrupts in a host/IOP system: those directed to the host processor, and those directed to the IOP Z-80A. The second category may be further subdivided into two groups: IOP interrupts from C-Bus peripherals, and IOP interrupts resulting from host I/O with the IOP registers, or **internal IOP interrupts**, to use the terminology adopted earlier in the manual.

IOP -> Host Interrupts

The IOP interrupts the host system processor by setting **Control Register** bit **Enable S-100 Interrupts**, and then writing a vector byte to **Interrupt Register.** The first action presents logic 1 data to latch IC24 at pin 12, and the output strobe from the second sets the same latch at pin 9. This in turn supplies current to the base of Q6, which then pulls S-100 line $\overline{\text{INT}}$ active low to issue the actual request. At the same time, latch output IC24 pin 9 is fed to the data input of latch IC34 at pin 2. This data is strobed at the beginning of the following M1 cycle to set the latch, and thus forces S-100 bus daisy chain line $\overline{\text{PRIORITY OUT}}$ active low at J2 pin 2. This signals lower priority tasks in the chain that the IOP is awaiting host interrupt service.

The IOP may be one of several tasks in an S-100 interrupt daisy chain. Interrupt acknowledge status from the host (sINTA = 1) is directed to the single task with its $\overline{\text{PRIORITY IN}}$ line inactive high (no higher priority tasks are requesting service) and its $\overline{\text{PRIORITY OUT}}$ line active low (the task at hand **is** requesting service). Host INTA directed to the IOP causes IC33 pin 8 to pulse low, which drives IC56 pins 1 and 19 active

low and so gates the contents of the **Interrupt Register** onto the S-100 Data In bus. It also resets latch output IC24 pin 9 to release S-100 bus line $\overline{INT}$ by turning Q6 off.

C-Bus Peripherals -> IOP Interrupts

C-Bus peripherals issue NMI interrupt requests to the IOP Z-80A by forcing C-Bus line $\overline{NMI}$ active low, and they issue maskable interrupt requests to the IOP by forcing C-Bus line $\overline{INT}$ active low. The IOP Z-80A is obliged to honor all NMI requests, but maskable interrupt requests are acknowledged with INTA status only when interrupts are enabled by the IOP software. When C-Bus peripherals assert line $\overline{INT}$, <44>, active low, they must also force C-Bus line $\overline{PRI\ 3}$, <48>, active low. This action inhibits the **Internal IOP Interrupt** circuitry, which has a lower C-Bus interrupt priority than external C-Bus peripheral interrupt requests, from responding to INTA status from the IOP Z-80A. The Z-80A signals INTA status by driving both its $\overline{M1}$ and $\overline{IORQ}$ lines low. The highest priority C-Bus peripheral requesting IOP interrupt service uses this event to gate its interrupt vector onto the C-Bus D0 - D7 lines, to remove its interrupt request, and to deactivate its $\overline{PRIORITY\ OUT}$ line.

Internal IOP Interrupts

Setting **Control Register** bit **Enable IOP Interrupts** presents a logic 1 to the data input of latch IC32 at pin 12. This latch is clocked at pin 11 shortly after the host system has either written to IOP ports **Command Register** or **Input Data**, or has read from IOP ports **Status Register** or **Output Data**. Latch output IC32 pin 9 is then set as a result, and the set level then signals that an internal IOP interrupt is pending. If C-Bus line $\overline{PRI\ 3}$ is inactive high at IC2 pin 15 (meaning no higher priority C-Bus peripherals are currently requesting interrupt service from the IOP), the pending internal request drives Z-80A pin $\overline{INT}$ active low by turning Q1 on at the beginning of the first M1 cycle which follows. If $\overline{PRI\ 3}$ is active low, the internal request is held off (IC2 pin 15 is held reset) until all C-Bus interrupts are serviced (when $\overline{PRI\ 3}$ goes inactive high).

Again, the Z-80A acknowledges internal interrupt requests by forcing $\overline{M1}$ and $\overline{IORQ}$ low in unison. This event then: (1) drives IC20 pin 12 low which gates interrupt vector FEh onto the internal IOP data bus by

momentarily turning transistor Q3 on (this forces D0 = 0 while all other data lines are passively pulled up to +5 VDC by RN3), and (2) removes the internal interrupt request by resetting latch IC32 at pin 13.

4.10 C-BUS INTERFACE

For the most part, the IOP C-Bus interface connects the Z-80A address, data and status lines to all C-Bus peripherals in parallel during non-DMA cycles, and disconnects (floats) the same lines during C-Bus DMA cycles (see Figure 28). IC22 pin 13 controls bidirectional data bus transceiver IC12 with signal INTERNAL as discussed in Section 4.7 above. The next manual section pictorially describes the waveforms which appear on these lines for each IOP Z-80A cycle type.

There are eight active C-Bus signal lines in addition to this group. $\overline{\text{NMI}}$, $\overline{\text{INT}}$ and $\overline{\text{WAIT}}$ are wire-AND lines with passive pull up resistors which directly drive Z-80A inputs of the same name. $\overline{\text{HALT}}$ is an always active buffered version of Z-80A status $\overline{\text{HALT}}$, and C-Bus ϕ is an **inverted**, always active buffered version of Z-80A clock input ϕ. $\overline{\text{CPU DISCONNECT}}$ is a software polled line whereby peripherals make requests for DMA control of the C-Bus, and $\overline{\text{BUS AVAILABLE}}$ is a software controlled DMA acknowledgment status line. Finally, $\overline{\text{PRI3}}$ is a hardware controlled line which manages external C-Bus peripheral, and internal IOP interrupt request priorities.

4.11 C-BUS WAVEFORMS

The figures below show the behavior of C-Bus signal lines during each IOP Z-80A cycle type. These figures are pattered after those which appear in the Z-80A Technical Manual. The Z-80A Technical Manual should be consulted for detailed timing specifications among these signals, but the propagation delay times of the C-Bus buffer/drivers must be included in any such analysis (tPHL and tPLH are 18 nSec maximum for most IOP buffer/drivers).

Figures 32 and 33 illustrate M1 (opcode fetch) cycles with no Wait States, and with two Wait States, respectively. Likewise, Figures 34 and 35 illustrate Memory Read and Memory Write cycles with no Wait States, and two Wait States. Figure 36 shows Input and Output cycles with no Wait States (the one Wait State shown is

automatically inserted by the Z-80A). Figure 37 shows Input and Output cycles with one inserted Wait State (the first is automatically generated by the Z-80A, and the other inserted by C-Bus line $\overline{WAIT}$). Figure 38 shows the C-Bus lines during maskable interrupt request and acknowledge cycles (IM0 mode response shown), and Figure 39 shows NMI request and acknowledge cycles. Finally, Figure 40 shows the C-Bus signal lines as the Z-80A enters a HALT state under software control, followed by an interrupt exit from the HALT state. Notice that the Z-80A internally executes NOPs (one such cycle shown) while in the HALT state.

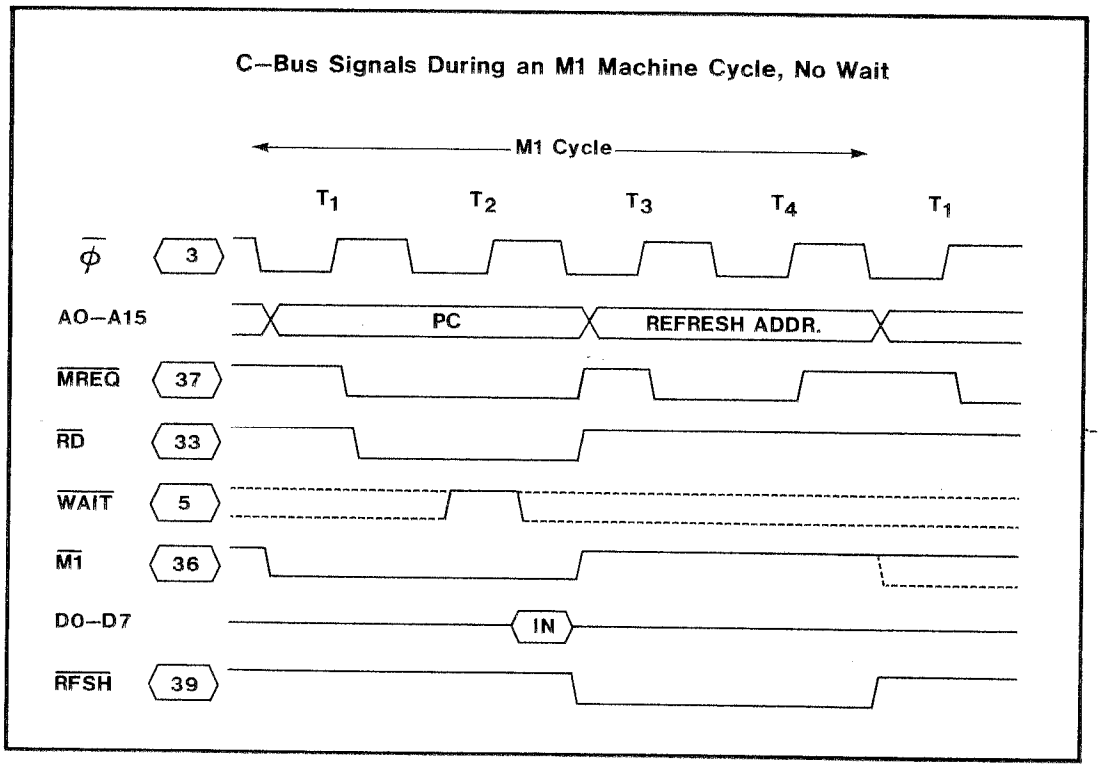


Figure 32: M1 CYCLE, NO WAIT STATES

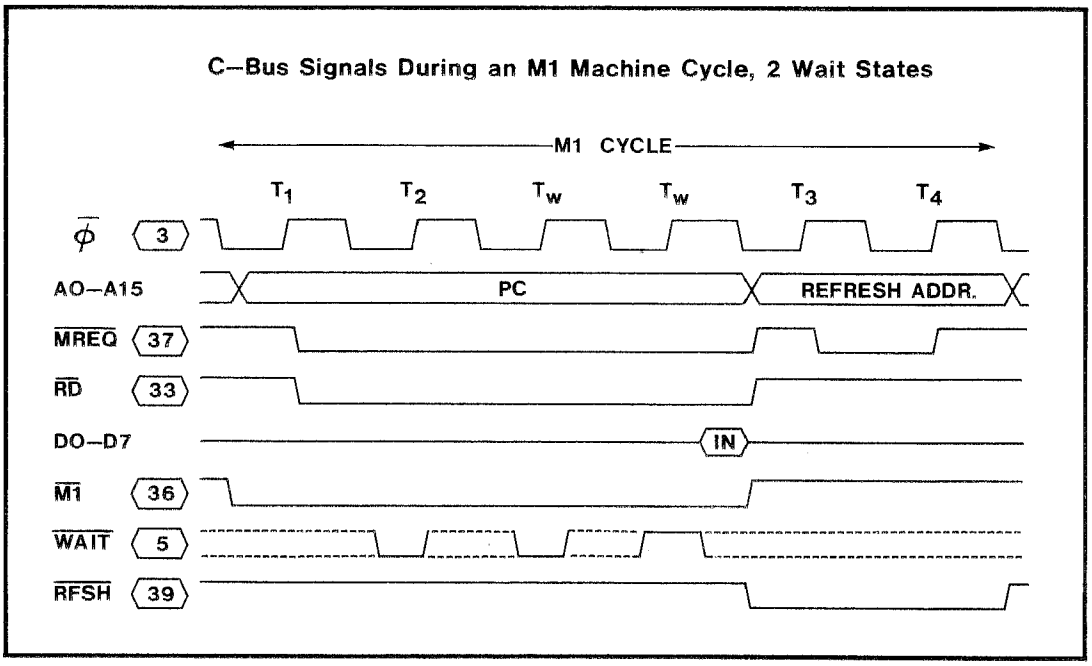


Figure 33: M1 CYCLE, 2 WAIT STATES

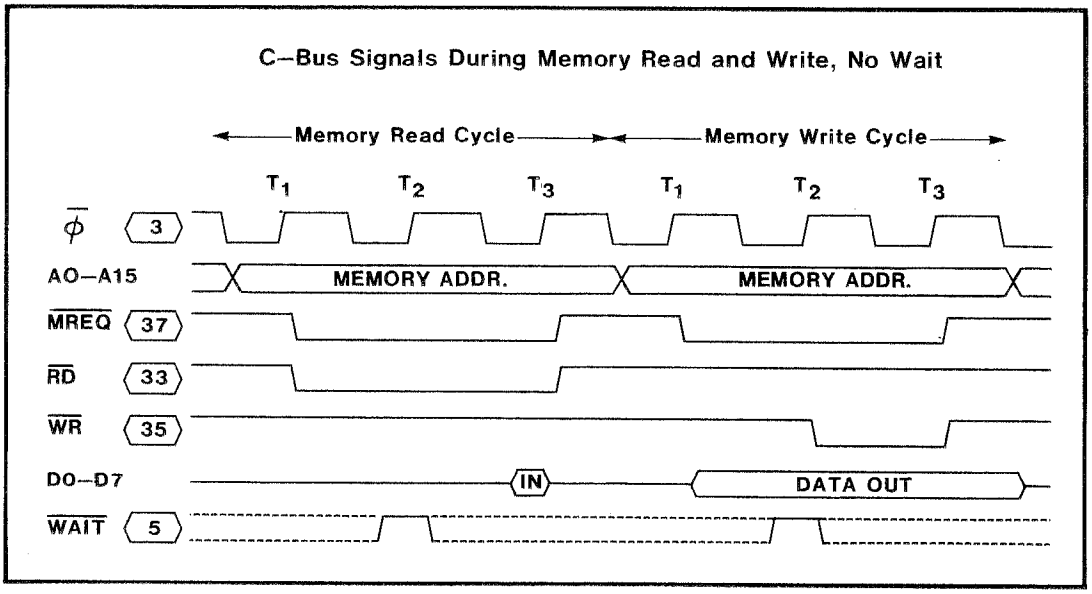


Figure 34: MEMORY READ/WRITE, NO WAIT STATES

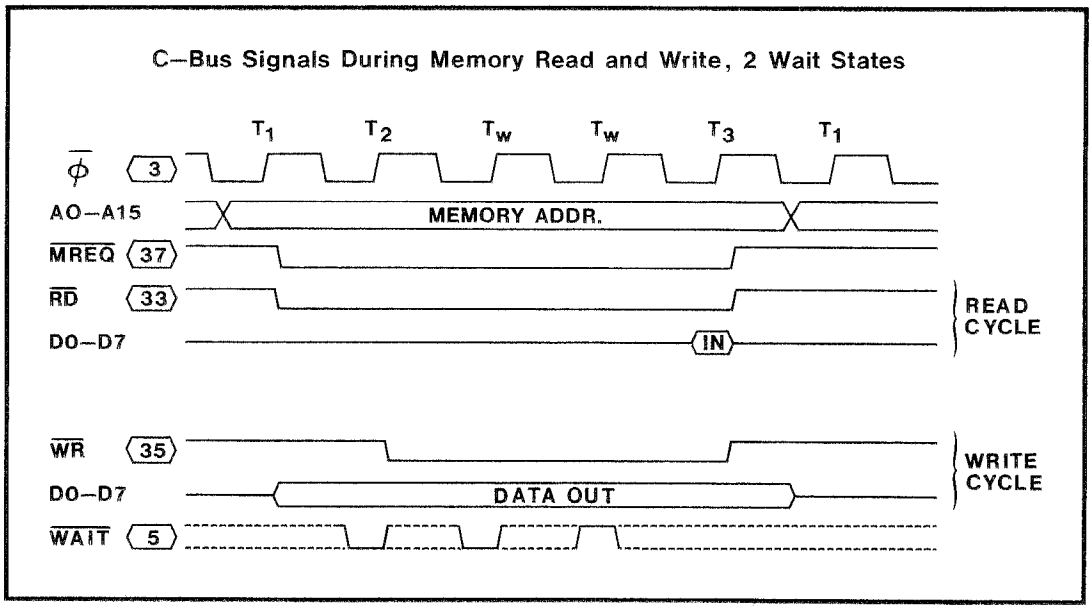


Figure 35: MEMORY READ/WRITE, 2 WAIT STATES

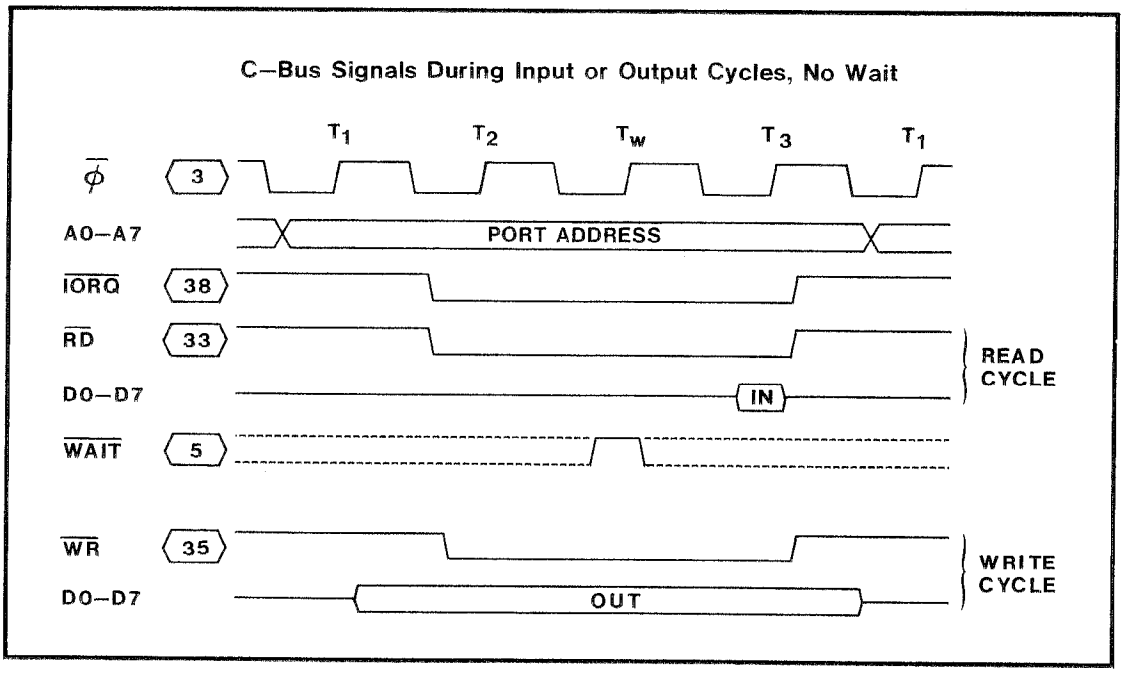


Figure 36: INPUT/OUTPUT, NO WAIT STATES

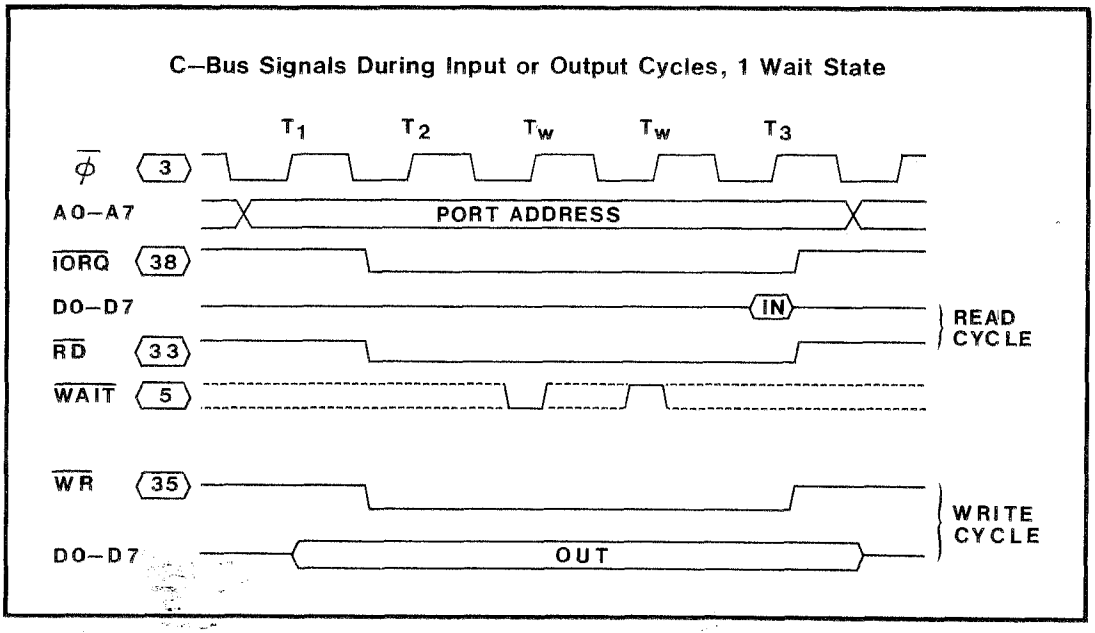


Figure 37: INPUT/OUTPUT, 1 WAIT STATE

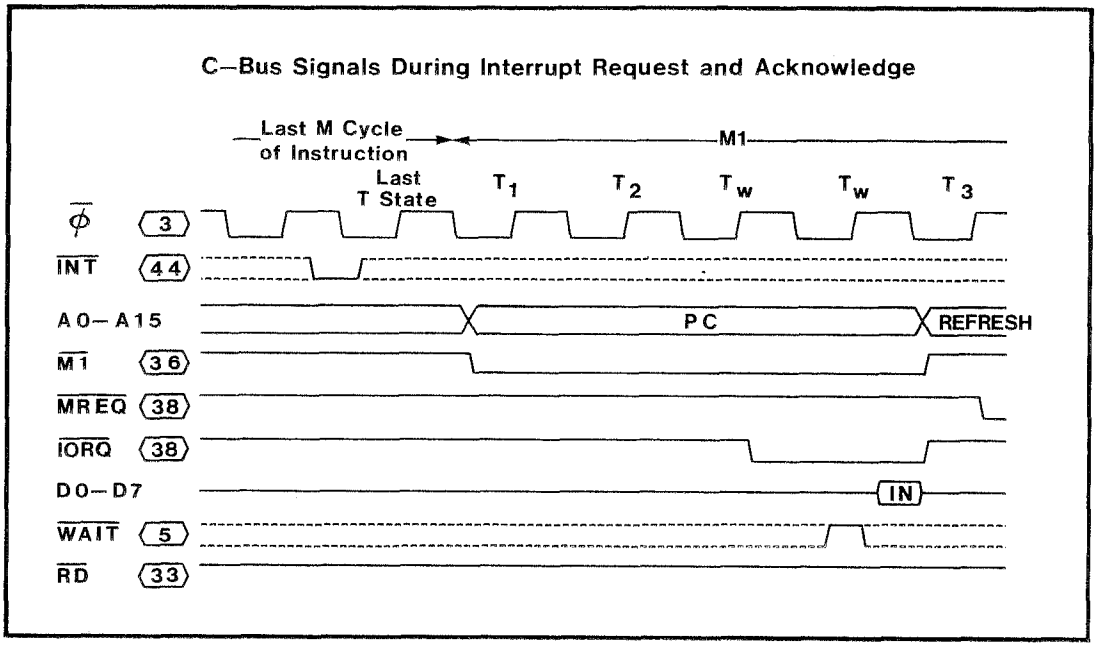


Figure 38: INTERRUPT REQUEST AND ACKNOWLEDGE

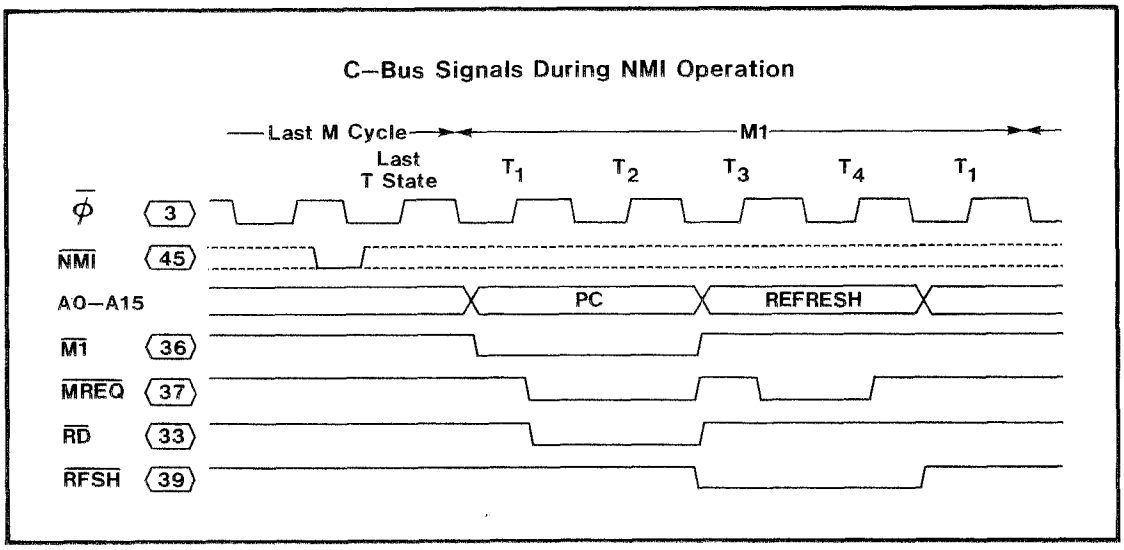


Figure 39: NMI CYCLE

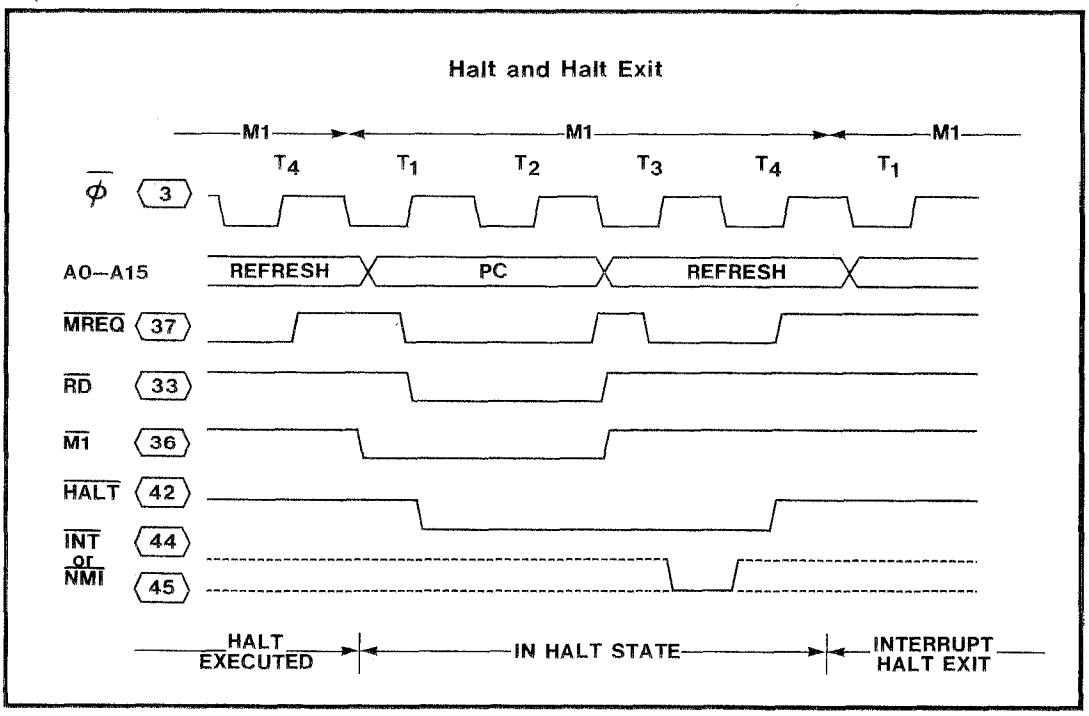


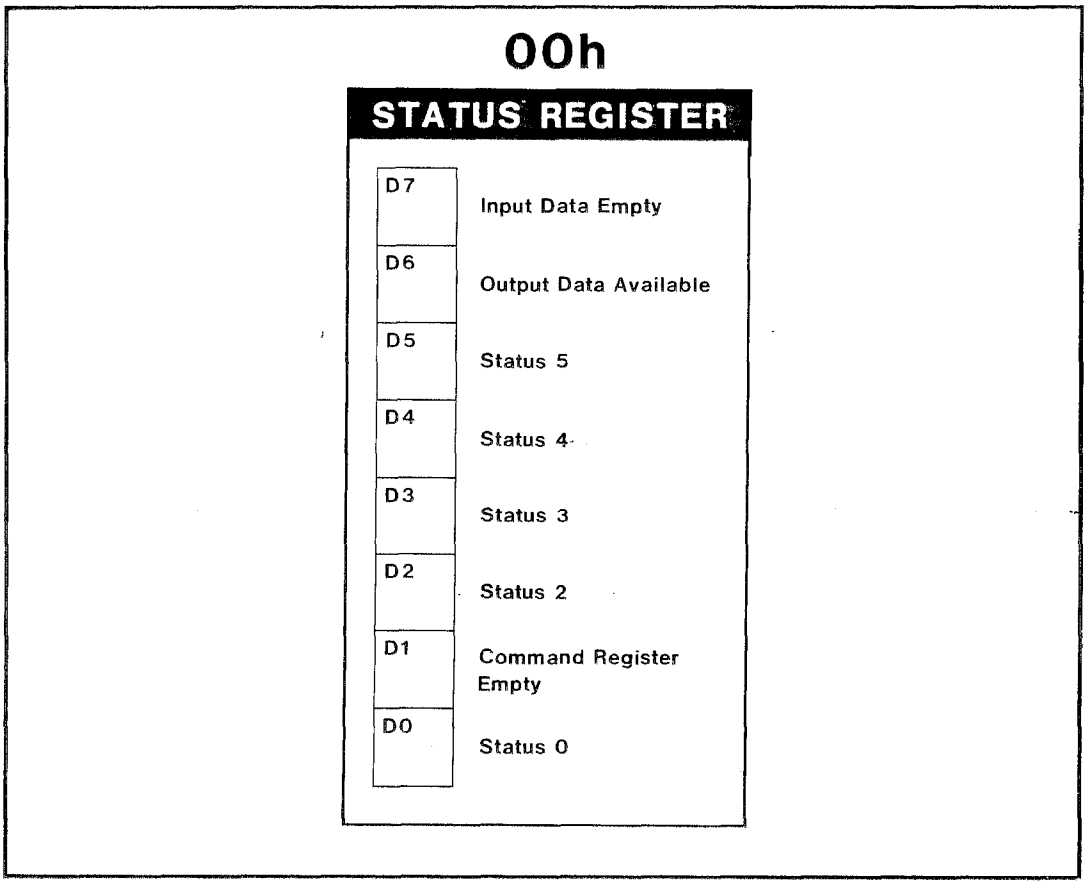
Figure 40: HALT AND HALT EXIT

Appendix A

IOP REGISTER BIT FUNCTION DESCRIPTIONS

STATUS REGISTER

IOP : OUT 00h
Host: IN Ibase+00h



This register supplies the host processor with handshake lines for exchanging data with the IOP, and it also allows the IOP to supply five software driven status bits to the host. A host read from this port may be programmed to generate an internal IOP interrupt by setting **Control Register** bit **Enable IOP Interrupts.**

D7 Input Data Empty

This bit is hardware controlled; data output by the IOP to this bit position is ignored. This bit is reset immediately after the host processor writes a data byte to register **Input Data.** This bit is set (signifying that the host may write another data byte to the IOP) as the IOP reads register **Input Data,** by a POC, or a C-Bus Reset.

D6 Output Data Available

This bit is hardware controlled; data output by the IOP to this bit position is ignored. This bit is set (signifying that an IOP written data byte is available for host reading) when the IOP writes a data byte to register **Output Data.** This bit is reset immediately after the host processor reads register **Output Data,** by a POC, or a C-Bus Reset.

D5 Status 5

This bit function is defined by the IOP software. The bit is reset by a POC or a C-Bus Reset.

D4 Status 4

This bit function is defined by the IOP software. The bit is reset by a POC or a C-Bus Reset.

D3 Status 3

This bit function is defined by the IOP software. The bit is reset by a POC or a C-Bus Reset.

D2 Status 2

This bit function is defined by the IOP software. The bit is reset by a POC or a C-Bus Reset.

D1 Command Register Empty

This bit is set (signifying that the host may write a new command byte to the IOP) when the IOP outputs a logic 1 to this bit position, or by a POC, or a C-Bus Reset. This bit is reset as the host processor writes a command byte to the **Command Register.** The IOP must set

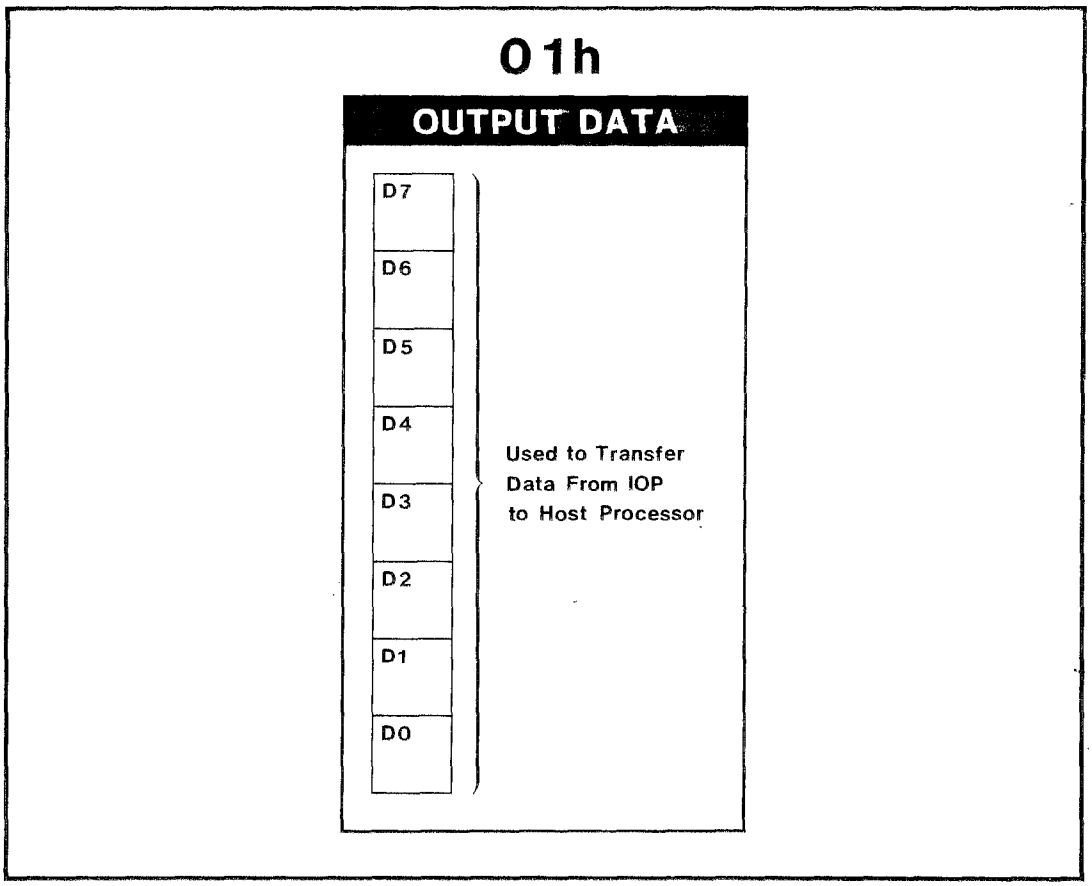
this bit for the host under IOP program control since the bit is **not** automatically set when the host reads the **Command Register.** Outputting a logic 1 to this bit position does set it, but it does not latch it set; outputting a logic 0 to this bit position is a null operation.

D0 Status 0

This bit function is defined by the IOP software. The bit is reset by a POC or a C-Bus Reset.

OUTPUT DATA

IOP : OUT 01h
Host: IN Ibase+01h

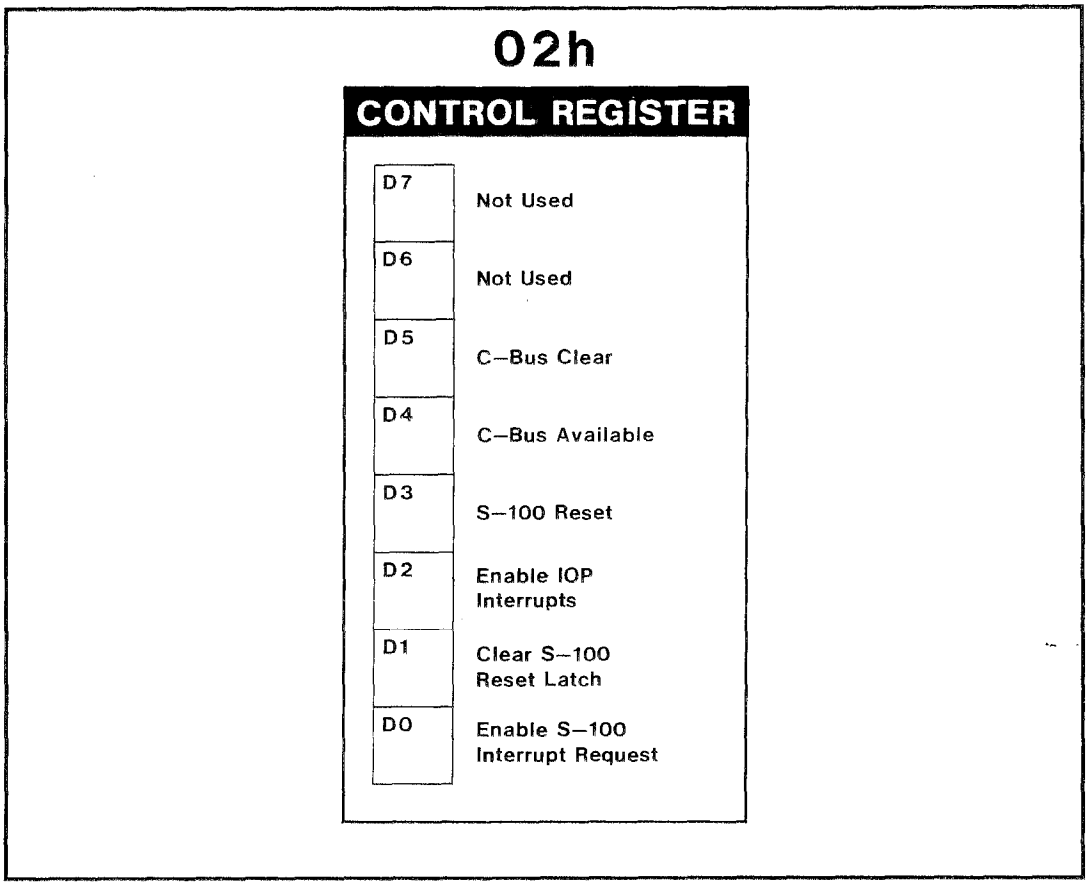


The IOP passes eight bits of parallel data to the host processor through this register. These bits are unaffected by a POC or a C-Bus Reset, and should be assumed random until after data is written to this port for the first time. **Status Register** bit **Output Data Available** is set to alert the host that a data byte from the IOP is available in this register for reading. When the host reads register **Output Data**, then **Flag Register** bit **Output Data Empty** is set to alert the IOP that it may output a new data byte. If the IOP writes data to this port before the host processor has read the previous byte, the new data overwrites the old; if the host reads register **Output Data** before new data is available, the previous contents are merely reread. A host read from this port may be programmed to generate an internal IOP interrupt by setting **Control Register**

bit **Enable IOP Interrupts.** The IOP may issue a host interrupt request after writing data to the host by writing a vector byte to register **Interrupt Vector** while **Control Register** bit **Enable S-100 Interrupt Request** is set.

CONTROL REGISTER

IOP : OUT 02h
Host: No Access



D7 Not Used

D6 Not Used

D5 C-Bus Clear

The IOP outputs a logic 1 to this bit position to pulse C-Bus line <2>, $\overline{\text{RESET}}$, active low for 625 nSec (nominal), and to preset IOP internal register bits to the states shown in Figure 39; the IOP Z-80A is **not** reset, however, so IOP program execution continues uninterrupted. Outputting a logic 0 to this bit position is a null operation.

D4 C-Bus Available

The IOP outputs a logic 1 to this bit position to latch C-Bus line $\overline{\text{BUS AVAILABLE}}$, <41>, active low, and to relinquish control of the C-Bus during DMA data transfers. This is typically done in response to a DMA request when **Flag Register** bit **CPU Disconnect** is polled set. While bit **C-Bus Available** is set, the IOP forces its address driver (A15 - A0), bidirectional data bus transceiver (D7 - D0) and status line driver ($\overline{\text{IORQ}}$, $\overline{\text{MREQ}}$, $\overline{\text{M1}}$, $\overline{\text{RD}}$, $\overline{\text{WR}}$, $\overline{\text{RFSH}}$ and $\overline{\text{RESET}}$) outputs to high impedance states allowing a DMA device to control these lines. Any running IOP program continues to run, and it would typically poll **Flag Register** bit **CPU Disconnect** until reset (C-Bus DMA transfer complete). The IOP then outputs a logic 0 to **C-Bus Available** to regain control of the C-Bus address, data and status lines. This bit is reset by a POC or a C-Bus Reset.

D3 S-100 Bus Reset

The IOP outputs a logic 1 to this bit position to force lines S-100 $\overline{\text{RESET}}$ and C-Bus $\overline{\text{RESET}}$ active low, and additionally to preset the IOP internal register bits to the states shown in Figure 39; this action is equivalent to a POC on both the S-100 and C-Busses. Note that setting this bit resets the IOP Z-80A which results in an immediate jump to IOP memory address 0000h. Either a POC or a C-Bus Reset forces bit **S-100 Bus Reset** to logic 0; consequently this bit is automatically reset after being set. Outputting a logic 0 to this bit position is a null operation.

D2 Enable IOP Interrupts

A maskable interrupt request is issued to the IOP Z-80A as the host either reads from, or writes to any IOP register when this bit is set (host inputs from **Status Register** or **Output Data**, or host outputs to **Command Register** or **Input Data**). This interrupt request is prioritized among any other pending C-Bus interrupt requests, and it is assigned the lowest C-Bus priority (below CBUS 2). Interrupt vector FEh, which anticipates either a Z-80A interrupt mode IM1 or IM2 response, is automatically placed on the internal IOP data bus during Interrupt Acknowledge. Resetting this bit inhibits interrupt requests from host read/writes (any pending request is removed on the next processor M1 cycle), but doing so does **not** inhibit interrupt requests from any other C-Bus peripheral (in this case CBUS 2 becomes the lowest priority interrupt source). This bit is automatically reset by either a POC or a C-Bus Reset.

D1 Clear S-100 Reset Latch

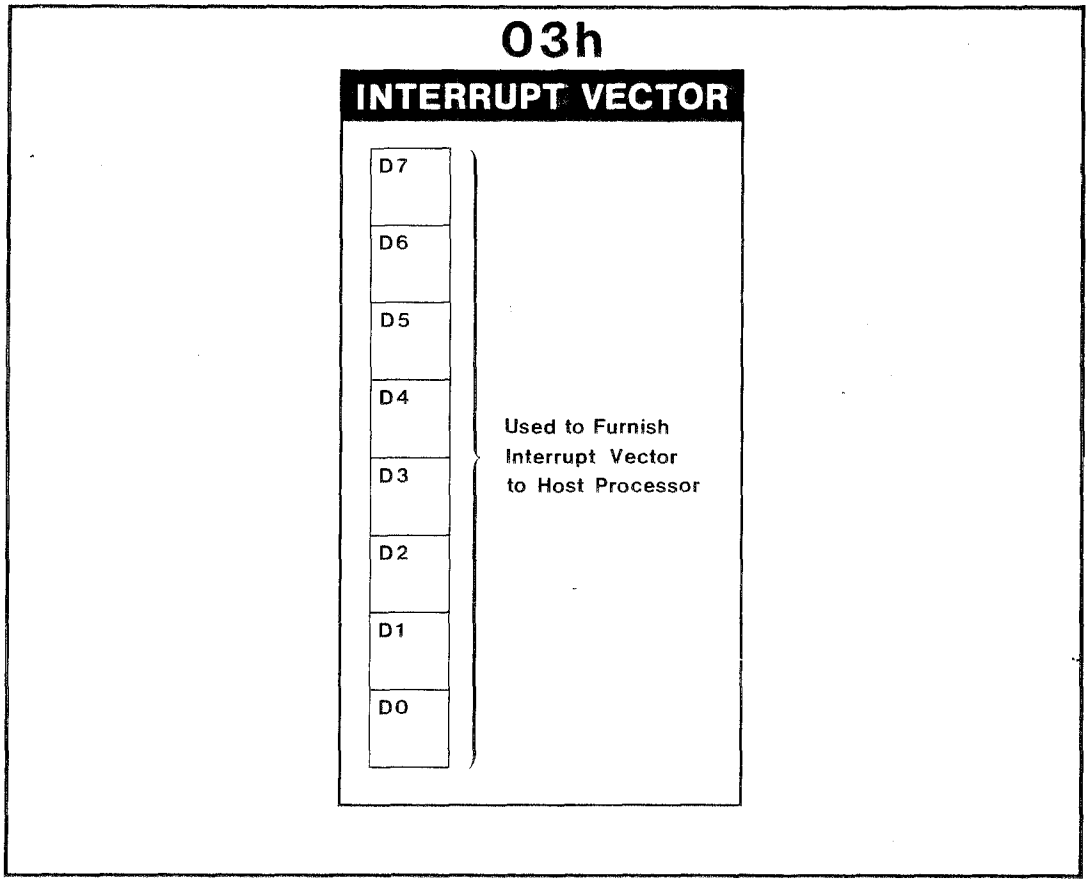
The IOP sets this bit to unconditionally latch **Flag Register** bit **Reset Request** to the logic 0 state. Resetting this bit frees **Flag Register** bit **Reset Request** to be set when an S-100 bus reset later occurs (when S-100 line $\overline{\text{RESET}}$ goes active low). This bit is reset by either a POC or a C-Bus Reset. If the IOP software is to actively monitor **Flag Register** bit **Reset Request**, then bit **Clear S-100 Reset Latch** should be first set, then reset, by the IOP initialization software.

D0 Enable S-100 Interrupt Request

The IOP sets this bit to arm circuitry which forces S-100 bus line $\overline{\text{INT}}$ active low each time the IOP loads a vector into register **Interrupt Vector**. The contents of register **Interrupt Vector** are then placed on the S-100 Data In bus when the host processor signals Interrupt Acknowledge directed to the IOP, and the IOP releases line $\overline{\text{INT}}$. Resetting this bit inhibits subsequent interrupt requests from the IOP to the host processor, but is does **not** remove any request which is pending at the time the bit is reset. Any such pending interrupt request is serviced normally. Either a POC or a C-Bus Reset bit **Enable S-100 Interrupt Request**, and they additionally remove any pending IOP interrupt request to the host.

INTERRUPT VECTOR

IOP : OUT 03h
Host: No Access

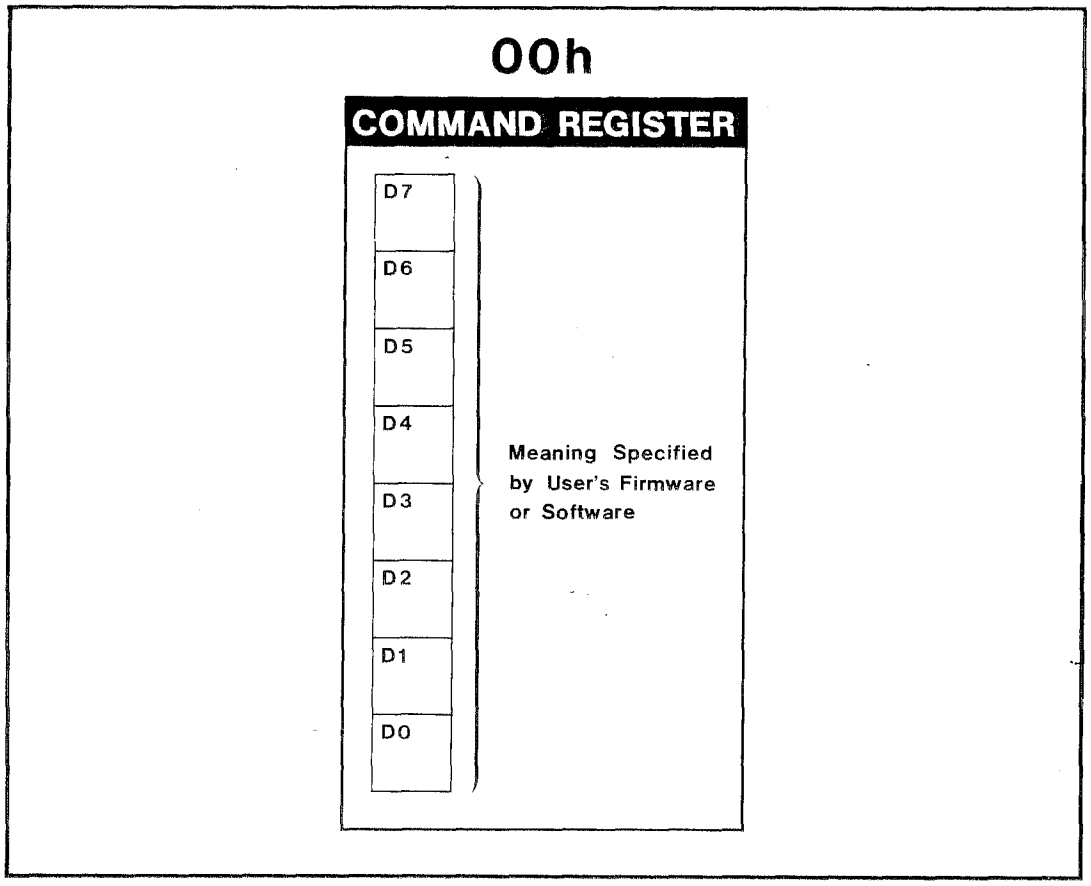


The IOP both issues a maskable interrupt request to the host processor, and defines the interrupt vector supplied during Interrupt Acknowledge from the host, by outputting a byte to this register while **Control Register** bit **Enable S-100 Interrupt Request** is set. If bit **Enable S-100 Interrupt Request** is reset, then IOP outputs to this register are null operations. If there is more than one S-100 interrupt source, then IOP interrupt requests are prioritized among the others with S-100 daisy chain interrupt cabling (IOP connector J2). If the host processor is operating in interrupt mode IM0, then the IOP would typically output the opcode corresponding to one of eight RST (restart) instructions to register **Interrupt Vector.** If operating in IM1, then the value output is irrelevant since the Z-80A defaults to host memory address 0038h for interrupt servicing.

If operating in IM2, then the IOP would output a byte which, when concatenated with the contents of the host's Z-80A I-register, would yield the indirect jump address of the service routine in host memory. The contents of this register are unaffected by a C-Bus Reset or POC, and are random until written to for the first time.

COMMAND REGISTER

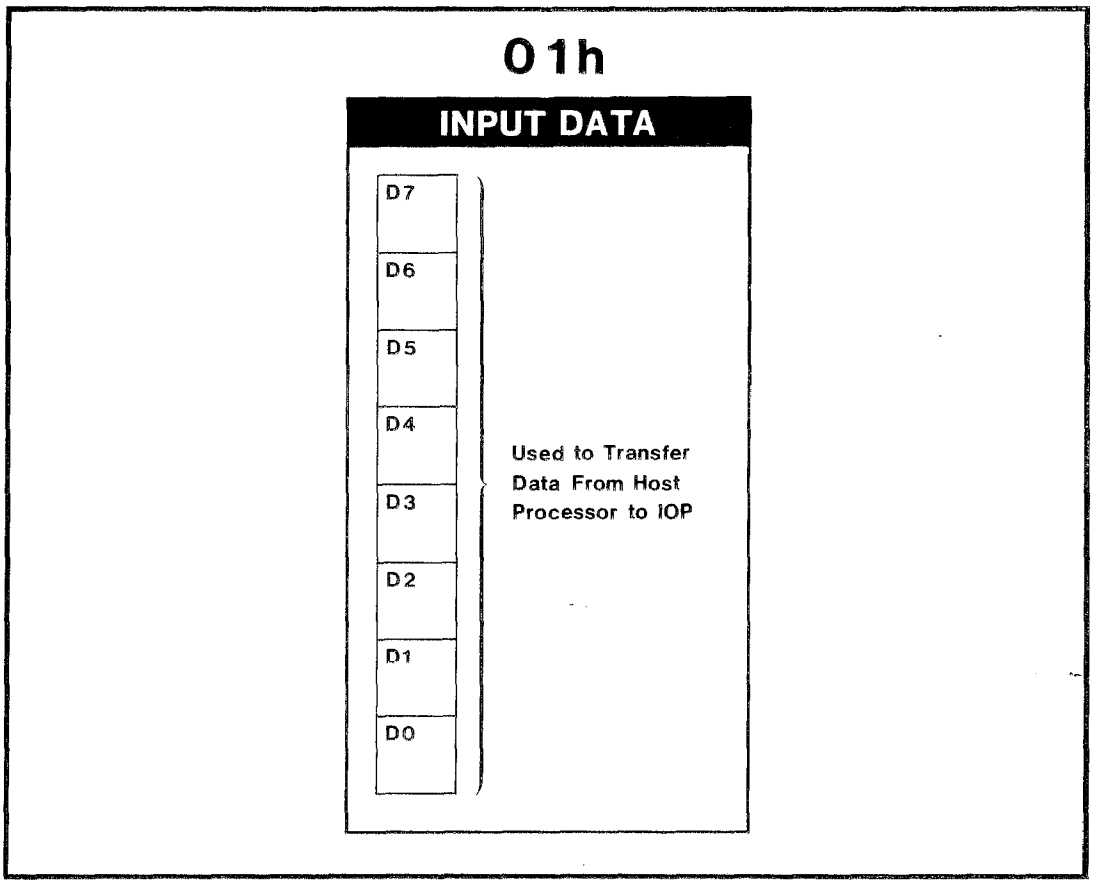
IOP : IN 00h
Host: OUT Ibase+00h



The host passes commands to the IOP through this port. The bits are user defined, so their interpretations depend on the IOP/Host software. These bits are unaffected by a POC or reset, and should be assumed random until a command is written to this port for the first time. **Flag Register** bit **Command Available** is set to alert the IOP that a command from the host is available in this register for reading. A host write to this port may be programmed to generate an internal IOP interrupt by setting **Control Register** bit **Enable IOP Interrupts.**

INPUT DATA

IOP : IN 01h
Host: OUT Ibase+01h

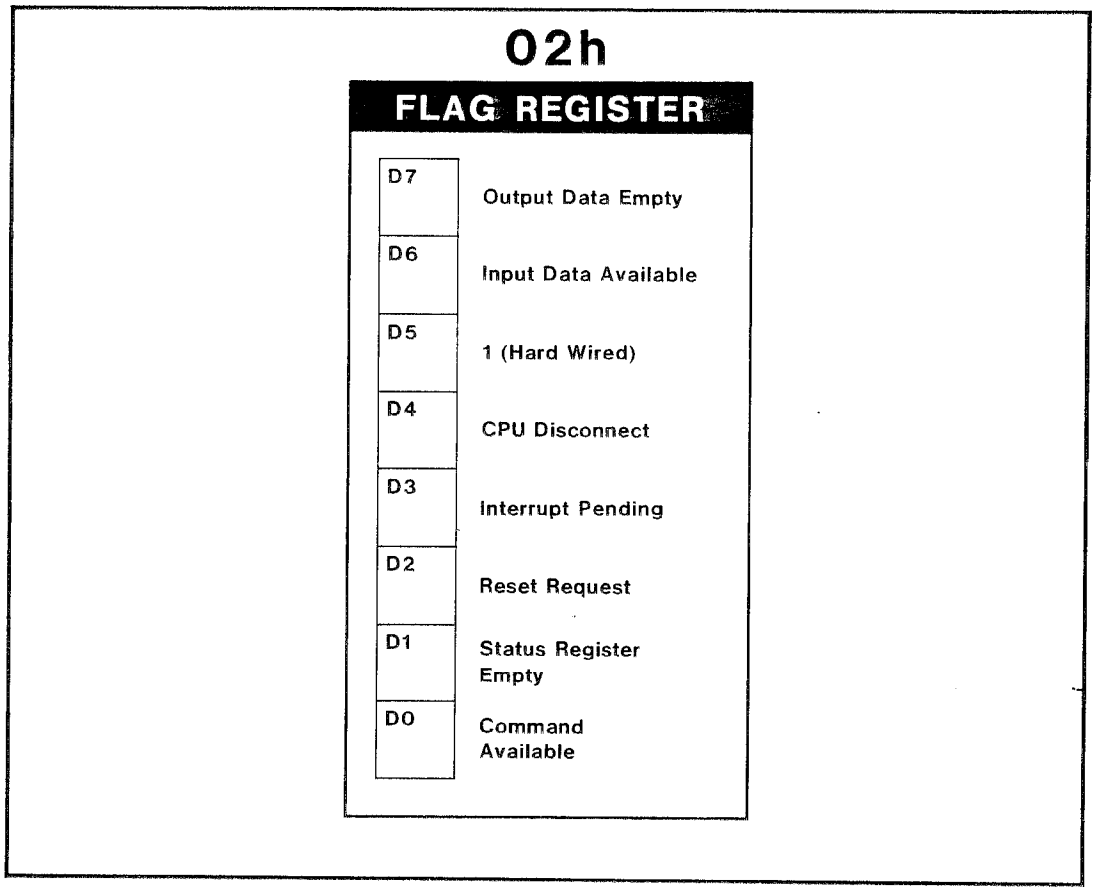


The host passes eight bits of parallel data to the IOP through this register. These bits are unaffected by a POC or a C-Bus Reset, and should be assumed random until data is written to this port for the first time. **Flag Register** bit **Input Data Available** is set to alert the IOP that a data byte from the host is available in this register for reading. When the IOP reads register **Input Data**, then **Status Register** bit **Input Data Empty** is set to alert the host that it may output a new data byte. If the host writes data to this port before the IOP has read the previous byte, the new data overwrites the old, and if the IOP reads register **Input Data** before new data is available, the previous contents are merely reread. A host write to this port may be programmed to generate an internal IOP interrupt by setting **Control Register** bit **Enable IOP Interrupts.** The IOP may issue a host

interrupt request after reading data from the host by writing a vector byte to register **Interrupt Vector** while **Control Register** bit **Enable S-100 Interrupt Request** is set.

FLAG REGISTER

IOP : IN 02h
Host: No Access



The IOP reads this register to determine the hardware status of its I/O registers and certain C-Bus and S-100 bus lines. All **Flag Register** bits are controlled by IOP hardware.

D7 Output Data Empty

This bit is reset as the IOP writes parallel data to its **Output Data** register. This bit is set (signifying that the IOP may write another data byte to the host processor) immediately after the host reads the **Output Data** register (this event may optionally be programmed to generate an internal IOP interrupt by setting **Control Register** bit **Enable IOP Interrupts**). Either a POC or a C-Bus Reset causes this bit to be set.

D6 Input Data Available

This bit is set (signifying that a host written data byte is available for IOP reading) immediately after the host has loaded a data byte into the **Input Data** register (this event may optionally be programmed to generate an internal IOP interrupt by setting **Control Register** bit **Enable IOP Interrupts**). This bit is reset as the IOP reads the **Input Data** register. Either a POC or a C-Bus Reset causes this bit to be reset.

D5 Logic 1

This bit is permanently set to logic 1.

D4 CPU Disconnect

This status mimics C-Bus line <40>, $\overline{\text{CPU DISCONNECT}}$; the bit is set when line <40> is active low, and reset when line <40> is inactive high. A C-Bus peripheral asserts line <40> active low to request control of the C-Bus for a DMA data transfer, and the IOP acknowledges the request by setting **Control Register** bit **C-Bus Available** after it has relinquished control of the C-Bus. After the DMA transfer is complete, the peripheral then must drive line <40> inactive high thereby resetting status bit **CPU Disconnect**. The IOP then uses this event to again seize control of the C-Bus.

D3 Interrupt Pending

This bit is set when the IOP is driving S-100 bus line $\overline{\text{INT}}$ active low, awaiting maskable interrupt servicing from the host processor. This bit is reset when the IOP is not driving line $\overline{\text{INT}}$ active low (other wire-ANDed devices may be forcing $\overline{\text{INT}}$ active low at this time, however). The IOP issues an interrupt to the host processor by: (1) setting **Control Register** bit **Enable S-100 Interrupt Request** which arms the interrupt circuitry, and then (2) loading a vector byte into register **Interrupt Vector**. Loading the vector byte automatically generates the interrupt request by forcing S-100 bus line $\overline{\text{INT}}$ active low. An Interrupt Acknowledge from the host directed to the IOP automatically removes the interrupt request, resetting bit **Interrupt Pending** in the process.

D2 Reset Request

This bit is unconditionally reset while **Control Register** bit **Clear S-100 Reset Latch** is latched set. This bit is set by a high to low transition on S-100 bus line $\overline{\text{RESET}}$ provided bit **Clear S-100 Reset Latch** is reset. The state of bit **Reset Request** should be assumed random after a POC or a C-Bus Reset. Since an active low level on S-100 bus line $\overline{\text{RESET}}$ does not reset any IOP circuitry, this status bit allows the IOP to detect a S-100 bus reset and optionally reset itself by setting **Control Register** bit **C-Bus Clear.**

D1 Status Register Empty

This bit is set immediately after the host reads the **Status Register** contents (this event may optionally be programmed to generate an internal IOP interrupt by setting **Control Register** bit **Enable IOP Interrupts).** This bit is reset as the IOP outputs a byte to the **Status Register.** This bit is set by a POC or a C-Bus Reset.

D0 Command Available

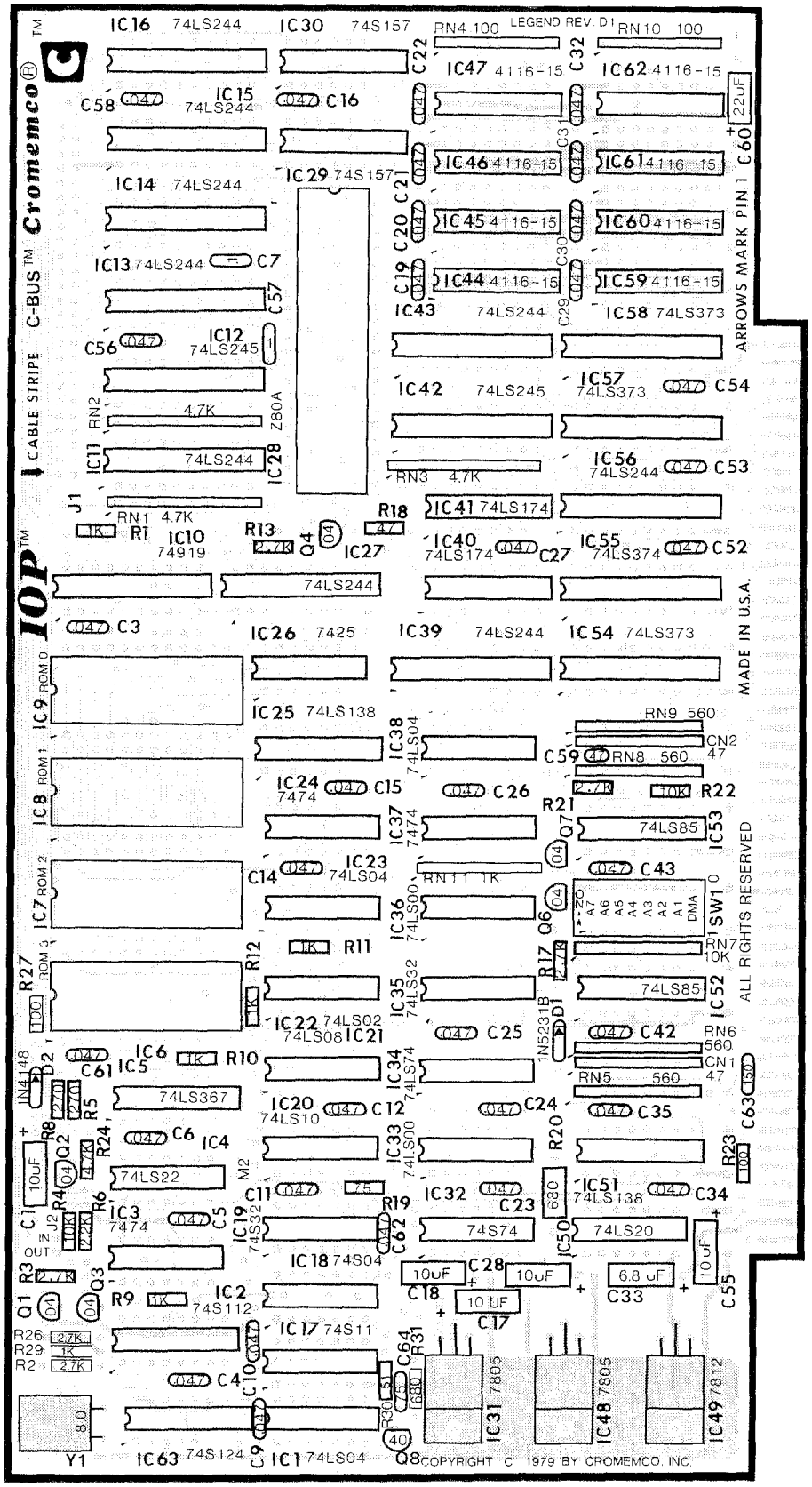
This bit is set as the host processor writes a byte to the **Command Register** (this event may optionally be programmed to generate an internal IOP interrupt by setting **Control Register** bit **Enable IOP Interrupts).** This bit is reset only when the IOP sets **Status Register** bit **Command Register Empty** under software control (note that the IOP cannot reset the **Command Available** bit by merely reading the **Command Register).** This bit is reset by a POC or a C-Bus Reset.

Parts List

Designations	Descriptions	Cromemco Part No.
Integrated Circuits		
IC1	(non TI) 74LS04	010-0066
IC2	74S112	010-0262
IC3	7474	010-0019
IC4	74LS22	010-0105
IC5	74LS367	010-0108
IC6-9	7416	not supplied
IC10	74919	502-0019
IC11	74LS244	010-0100
IC12	74LS245	010-0120
IC13-16	74LS244	010-0100
IC17	74S11	010-0111
IC18	74S04	010-0123
IC19	74S32	010-0090
IC20	74LS10	010-0063
IC21	74LS08	010-0064
IC22	74LS02	010-0068
IC23	(non TI) 74LS04	010-0066
IC24	7474	010-0019
IC25	74LS138	010-0096
IC26	7425	010-0032
IC27	74LS244	010-0100
IC28	Z80A/3880P-44	011-0010
IC29-30	74S157	010-0157
IC31	7805/340T-5	012-0001
IC32	74S74	010-0142
IC33	74LS00	010-0069
IC34	74LS74	010-0055
IC35	74LS32	010-0058
IC36	74LS00	010-0069
IC37	7474	010-0019
IC38	(non TI) 74LS04	010-0066
IC39	74LS244	010-0100
IC40-41	74LS174	010-0097
IC42	74LS245	010-0120
IC43	74LS244	010-0100
IC44-47	TMS-4116	011-0019
IC48	7805/340T-5	012-0001
IC49	7812	012-0002
IC50	74LS20	010-0095
IC51	74LS138	010-0096
IC52-53	74LS85	010-0053
IC54	74LS373	010-0102
IC55	74LS374	010-0133
IC56	74LS244	010-0100

Designations	Descriptions	Cromemco Part No.
IC57-58	74LS373	010-0102
IC59-62	TMS-4116	011-0019
Transistors Diodes		
Q1	2N3646	009-0000
Q2	2N3904	009-0001
Q3-4	2N3646	009-0000
Q5		
Q6-7	2N3646	009-0000
D1	1N5231B	008-0006
Capacitors		
C1	10 uf tant	004-0032
C2	30 pf disk	004-0003
C3-6	.047 uf axial	004-0061
C7	.1 uf disk	004-0030
C8		
C9-12	.047 uf axial	004-0061
C13		
C14-16	.047 uf axial	004-0061
C17-18	10 uf tant	004-0032
C19-27	.047 uf axial	004-0061
C28	10 uf tant	004-0032
C29-32	.047 uf axial	004-0061
C33	6.8 uf tant	004-0034
C34-35	.047 uf axial	004-0061
C36-41		
C42-43	.047 uf axial	004-0061
C44-51		
C52-54	.047 uf axial	004-0061
C55	100 uf tant	004-0032
C56	.047 uf axial	004-0061
C57	.1 uf disk	004-0030
C58	.047 uf axial	004-0061
C59		
C60	22 uf tant	004-0028
C61	.047 uf axial	004-0061
C at R14	.001 uf disk	004-0043
Capacitor Networks		
CN1-2	47 pf, 8 pin	005-0000

Designations	Descriptions	Cromemco Part No.
Resistor		
R1-3	1 K	001-0018
R4	10K	001-0030
R5	270	001-0011
R6	2.2 K	001-0021
R7	330	001-0012
R8	270	001-0011
R9	1 K	001-0018
R16		
R17	1 K	001-0018
R18	47	001-0003
R19	74	001-0006
R20 1/2 Watt	680	001-0067
R21	1 K	001-0018
R22	10K	001-0030
R23	270	001-0011
R across IC39	1 K	001-0018
Resistor Networks		
RN1-3	4.7K, 10 pin	003-0014
RN4	100, 8 pin	003-0001
RN5-6	560, 8 pin	003-0006
RN7	10K, 8 pin	003-0025
RN8-9	560, 8 pin	003-0006
RN10	100, 8 pin	003-0001
Miscellaneous		
	1 switch, 8 pos	013-0002
	1 connector, 50 pin	017-0014
	1 socket, 40 pin	017-0006
	4 socket, 24 pin	017-0005
	16 socket, 20 pin	017-0004
	18 socket, 16 pin	017-0002
	20 socket, 14 pin	017-0001
	1 socket 2 pin	017-0009
	1 crystal, 8 mhz	026-0001
	1 heatsink	016-0108
	1 bus bar, 28 pin	021-0097


Cromemco®
IOP™
C-BUS™
CABLE STRIPE
LEGEND REV. D1
ARROWS MARK PIN 1
MADE IN U.S.A.
ALL RIGHTS RESERVED
IC16 74LS244
IC30 74S157
RN4 100
RN10 100
IC47 4116-15
IC62 4116-15
IC15 74LS244
IC14 74LS244
IC29 74S157
IC46 4116-15
IC61 4116-15
IC45 4116-15
IC60 4116-15
IC13 74LS244
IC44 4116-15
IC59 4116-15
IC43 74LS244
IC58 74LS373
IC12 74LS245
IC42 74LS245
IC57 74LS373
RN2 4.7K
IC11 74LS244
IC28 Z80A
RN3 4.7K
IC56 74LS244
J1
RN1 4.7K
IC41 74LS174
IC10 74919
IC27
IC40 74LS174
IC55 74LS374
74LS244
IC26 7425
IC39 74LS244
IC54 74LS373
IC9 ROM 0
IC25 74LS138
IC38 74LS04
RN9 560
CN2
RN8 560
IC8 ROM 1
IC24 7474
IC37 7474
74LS85
IC53
IC7 ROM 2
IC23 74LS04
RN11 1K
IC36 74LS00
SW1
RN7 10K
IC27 ROM 3
IC35 74LS32
IC52
IC22 74LS02
74LS08
IC21
IN5231B
IC6
IC5
IC34 74LS74
RN6 560
CN1
RN5 560
74LS367
IC20 74LS10
IC12
IC4
IC33 74LS00
74LS22
IC32
IC51 74LS138
IC3 7474
IC19 74S32
74S74
IC50 74LS20
IC18 74S04
IC2 74S112
IC17 74S11
IC63 74S124
IC1 74LS04
Q8
IC31 7805
IC48 7805
IC49 7812
Y1
COPYRIGHT © 1979 BY CROMEMCO, INC.

LIMITED WARRANTY

Cromemco, Inc. ("Cromemco") warrants this product against defects in material and workmanship to the original purchaser for ninety (90) days from the date of purchase, subject to the following terms and conditions.

What Is Covered By This Warranty

During the ninety (90) day warranty period Cromemco will, at its option, repair or replace this Cromemco product or repair or replace with new or used parts any parts or components, manufactured by Cromemco, which prove to be defective, provided the product is returned to an Authorized Cromemco Dealer as set forth below.

How To Obtain Warranty Service

You should immediately notify IN WRITING your Authorized Cromemco Dealer or Cromemco Inc of problems encountered during the warranty period. In order to obtain warranty service, first obtain a return authorization number by contacting the Authorized Cromemco Dealer from whom you purchased the product. Then attach to the product:

1. Your name, address and telephone number,
2. the return authorization number,
3. a description of the problem, and
4. proof of the date of retail purchase.

Ship or otherwise return the product, transportation and insurance costs prepaid, to the Authorized Cromemco Dealer. If you are unable to receive warranty repair from the Authorized Cromemco Dealer from whom you purchased the product, you should contact Cromemco Customer Support at: Cromemco, Inc., 280 Bernardo Ave., Mountain View, Ca. 94043.

What Is Not Covered By This Warranty

Cromemco does not warrant any products, components or parts not manufactured by Cromemco.

This warranty does not apply if the product has been damaged by accident, abuse, misuse, modification or misapplication; by damage during shipment; or by improper service. This product is not warranted to operate satisfactorily with peripherals or products not manufactured by Cromemco. Transportation and insurance charges incurred in transporting the product to and from the Authorized Cromemco Dealer or Cromemco are not covered by this Warranty.

THIS WARRANTY IS IN LIEU OF ALL OTHER WARRANTIES, WHETHER ORAL OR WRITTEN, EXPRESS OR IMPLIED. ANY IMPLIED WARRANTIES, INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A-PARTICULAR PURPOSE, ARE LIMITED IN DURATION TO NINETY (90) DAYS FROM THE DATE OF PURCHASE OF THIS PRODUCT. CROMEMCO SHALL NOT BE LIABLE FOR INCIDENTAL AND/OR CONSEQUENTIAL DAMAGES FOR THE BREACH OF ANY EXPRESS OR IMPLIED WARRANTY, INCLUDING DAMAGE TO PROPERTY AND, TO THE EXTENT PERMITTED BY LAW, DAMAGES FOR PERSONAL INJURY, EVEN IF CROMEMCO HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES. SOFTWARE, TECHNICAL INFORMATION AND FIRMWARE IS LICENSED "AS IS" AND WITH ALL FAULTS. THE AGENTS, DEALERS, AND EMPLOYEES OF CROMEMCO ARE NOT AUTHORIZED TO MAKE MODIFICATIONS TO THIS WARRANTY, OR ADDITIONAL WARRANTIES BINDING ON CROMEMCO ABOUT OR FOR PRODUCTS SOLD OR LICENSED BY CROMEMCO. ACCORDINGLY, ADDITIONAL STATEMENTS WHETHER ORAL OR WRITTEN EXCEPT SIGNED WRITTEN STATEMENTS FROM AN OFFICER OF CROMEMCO DO NOT CONSTITUTE WARRANTIES AND SHOULD NOT BE RELIED UPON.

SOME STATES DO NOT ALLOW THE EXCLUSION OR LIMITATION OF INCIDENTAL OR CONSEQUENTIAL DAMAGES OR LIMITATIONS ON HOW LONG AN IMPLIED WARRANTY LASTS, SO THE ABOVE LIMITATION OR EXCLUSION MAY NOT APPLY TO YOU.

THIS WARRANTY AND THE STATUTE OF LIMITATIONS SHALL RUN CONCURRENTLY WITH ANY ACCEPTANCE PERIOD. THIS WARRANTY IS NOT TRANSFERABLE. NO SUIT, LITIGATION, OR ACTION SHALL BE BROUGHT BASED ON THE ALLEGED BREACH OF THIS WARRANTY OR IMPLIED WARRANTIES MORE THAN ONE YEAR AFTER THE DATE OF PURCHASE IN THOSE JURISDICTIONS ALLOWING SUCH A LIMITATION, OTHERWISE NO SUCH ACTION SHALL BE BROUGHT MORE THAN ONE YEAR AFTER THE EXPIRATION OF THIS WARRANTY.

THIS WARRANTY SHALL NOT BE APPLICABLE TO THE EXTENT THAT ANY PROVISION OF THIS WARRANTY IS PROHIBITED BY ANY FEDERAL, STATE OR MUNICIPAL LAW WHICH CANNOT BE PREEMPTED. THIS WARRANTY GIVES YOU SPECIFIC LEGAL RIGHTS, AND YOU MAY ALSO HAVE OTHER RIGHTS WHICH VARY FROM STATE TO STATE.

5/81

Cromemco™
incorporated
Tomorrow's Computers Today
280 BERNARDO AVE., MOUNTAIN VIEW, CA 94043

023-6022