I/O Subsystem Models C and D System Programmer Reference Manual

CSM-1009-000

Cray Proprietary

Cray Research, Inc.

The CRAY X-MP EA computer system is exempt from the technical requirements of the FCC's Part 15 Subpart J rules pursuant to Section 15.801 (C).

Requests for copies of Cray Research, Inc. publications should be directed to:

CRAY RESEARCH, INC.
DISTRIBUTION
2360 Pilot Knob Road
Mendota Heights, MN 55120
Telephone (612) 452-6650

Comments about this publication should be directed to:

CRAY RESEARCH, INC.
HARDWARE PUBLICATIONS
770 Industrial Blvd.
Chippewa Falls, WI 54729

Publication Number: CSM-1009-000

Publication Title: I/O Subsystem Models C and D System Programmer Reference Manual

Remove	Insert	Page Numbers	# of Pages	General contents of manual described and changes (if any) listed
	★			This manual has been converted from a hardware reference manual (HR-00081) to a system programmer reference manual (CSM-1009-000) to protect Cray proprietary information. I/O Subsystem Model D (IOS-D) information has been incorporated. The material in this manual has been reorganized. Appendix A is now part of Section 7; Appendix B is now incorporated throughout Section 4; Appendix C is now Section 6; Appendix E is now Section 8; Appendix F is now a part of Section 4; Appendix G is now Section 5. This printing makes HR-00081 obsolete.

Record of Revision

Each time this manual is revised and reprinted, all changes issued against the previous version are incorporated into the new version, and the new version is assigned an alphabetic level which is indicated in the publication number on each page of the manual.

Changes to part of a page are indicated by a change bar in the margin directly opposite the change. A change bar in the footer indicates that most, if not all, of the page is new. If the manual is rewritten, the revision level changes but the manual does not contain change bars.

REVISION	DESCRIPTION
	September 1985 – Original printing.
	April 1989 – I/O Subsystem Model D (IOS-D) information has been incorporated. The material in this manual has been reorganized. Appendix A is now part of Section 7; Appendix B is now incorporated throughout Section 4; Appendix C is now Section 6; Appendix E is now Section 8; Appendix F is now a part of Section 4; Appendix G is now Section 5. The manual has been converted from a hardware reference manual (HR-00081) to a system programmer reference manual to protect Cray proprietary information. When ordering this manual, use publication number CSM-1009-000.

PREFACE

This manual describes the operation of the Cray Research, Inc. (CRI) I/O subsystem model C (IOS-C) and the I/O subsystem model D (IOS-D), which control external communication and mass storage for Cray computer systems.

An IOS-C or IOS-D contains up to four I/O processors (IOPs) and has either 4, 8, or 32 million 64-bit words of buffer memory. The IOS interfaces to all components of a Cray computer system through various channel interfaces.

This manual describes the IOPs, the buffer memory, and the channel interfaces to the IOPs. It also describes instructions, instruction formats, and function codes for most peripheral devices.

AUDIENCE

This manual is written to assist programmers and field engineers. The reader should be familiar with digital computers.

ORGANIZATION

This manual is organized as follows:

SECTION 1 - I/O SUBSYSTEM OVERVIEW - This section describes IOS-C and IOS-D differences and gives an overview of the IOPs, IOP functions and channel interfaces, and the IOS buffer memory.

SECTION 2 - I/O PROCESSOR - This section describes the IOP hardware, including the computation and control sections, I/O channels, and local memory.

SECTION 3 - BUFFER MEMORY - This section describes buffer memory speed, organization, access, addressing, and error protection.

SECTION 4 - I/O PROCESSOR CHANNEL INTERFACES - This section describes the channel interfaces and the function codes that control them.

SECTION 5 - BUFFER MEMORY/CENTRAL MEMORY BYPASS MODE - This section describes bypass mode operation for the buffer memory and central memory channels.

SECTION 6 - I/O CHANNEL ASSIGNMENTS - This section lists recommended channel assignments for all I/O channels in all IOPs.

SECTION 7 - IOP PROCESSOR INSTRUCTION SET - This section describes all instructions in the IOP instruction set.

SECTION 8 - IOS DIAGNOSTIC MODES - This section describes IOS diagnostic procedures.

NOTATIONAL CONVENTIONS

This manual uses the following notational conventions:

- Register bit positions are numbered from right to left as powers of 2, starting with bit 2^0.
- All numbers are decimal unless otherwise indicated. Octal numbers are indicated with a subscript 8 (for example 12_8).
- Additional notational conventions are used in instruction descriptions. These conventions are defined in Section 7.

RELATED PUBLICATIONS

The following manuals describe CRI mainframes that use the IOS-C and IOS-D:

- CMM-0404-0A0 — The *CRAY Y-MP Series Theory of Operations Manual* describes the features and internal operations of the CRAY Y-MP computer system. This manual describes the operation of the central processing units (CPUs), which run instructions, provide memory protection, report hardware exceptions, and provide interprocessor communications within the computer system. Central memory and I/O channels are also described. This manual is written for CRI field engineers and systems test personnel maintaining the computer. The reader should be familiar with digital computers and the basic architecture of the CRAY Y-MP computer system.
- CSM0110000 — The *CRAY X-MP/2 System Programmer Hardware Reference Manual* describes the functions of a CRAY X-MP dual-processor system. This manual describes the following features: system conventions, CPU resources, CPU control, CPU computing, and CPU instructions relating to a CRAY X-MP/2 computer system. This manual is written for CRI field engineers.
- CSM0111000 — The *CRAY X-MP/1 System Programmer Reference Manual* describes the functions of a CRAY X-MP single-processor system. This manual describes the following features: system conventions, CPU resources, CPU control, CPU computing, and CPU instructions relating to a CRAY X-MP/1 computer system. This manual is written for CRI field engineers.
- CSM0112000 — The *CRAY X-MP/4 System Programmer Reference Manual* describes the functions of a CRAY X-MP four-processor system. This manual covers the following areas: system conventions, CPU resources, CPU control, CPU computing, and CPU instructions relating to a CRAY X-MP/4 computer system. This manual is written for CRI field engineers.

- CSM-0400-000 The *CRAY Y-MP System Programmer Reference Manual* is a detailed architectural overview of the CRAY Y-MP mainframe and is written to help system programmers write and optimize program code. This manual contains a detailed description of Cray assembly language (CAL) including hold issue conditions, execution times, and special cases. The manual contains the following sections: CRAY Y-MP Computer System Overview, Shared Resources, CPU Control, CPU Computation, Parallel Processing Features, Maintenance Mode, and CPU Instruction Descriptions.

- HR-00029 The *CRAY-1 S Series Computer Systems Mainframe Reference Manual* describes the functions of CRAY-1 S series computer systems, the overall computer system, its configurations, and equipment. It also describes the operation of the CPU, which runs programs and user jobs, and oversees job flow within the CRAY-1 S series computer system.

- HR-00064 The *CRAY-1 M Series Mainframe Reference Manual* describes the functions of the CRAY-1 M series computer systems. It is written for programmers and CRI field engineers with the assumption that the reader is familiar with digital computers. This manual describes the operation of the CPU and contains detailed reference information in the appendices.

- HR-00077 The *Disk Systems Hardware Reference Manual* describes the operation of the CRI DCU-4 and DCU-5 disk controller units, and the DD-29, DD-39, and DD-49 disk storage units. This manual is written for CRI field engineers and programmers with the assumption that the reader is familiar with digital computers as well as programming the Cray I/O subsystem (IOS). Descriptions of disk storage unit operation and appendices with detailed reference information are provided.

- HR-04001 A The *CRAY Y-MP Computer Systems Functional Description Manual* describes all components of the CRAY Y-MP computer system. The manual contains the following tabbed sections: System Overview, Mainframe Architecture and CPU Instructions, I/O Subsystem, SSD Solid-state Storage Device, Peripheral Equipment, and Software Overview. Specification sheets are included for the mainframe, IOS, SSD, and peripheral equipment sections. These sheets describe each model in detail. A glossary and index are provided at the end of the manual. This manual is written primarily for customers; a secondary audience includes CRI personnel requiring an overview of the CRAY Y-MP computer system.

CONTENTS

STANDARD IOP CHANNELS (continued)

FIGURES

FIGURES (continued)

TABLES

TABLES (continued)

1 - I/O SUBSYSTEM OVERVIEW

The Cray Research, Inc. (CRI) I/O subsystem (IOS) provides high-capacity data communications between the central memory of a CRI mainframe and peripheral devices such as data storage devices and front-end computers. The IOS model C (IOS-C) is a required component of CRAY-1 S series computer systems, SN/1200 through SN/4400, and all CRAY X-MP and CRAY-1 computer systems. The IOS model D (IOS-D) is a required component of the CRAY Y-MP computer systems. The IOS chassis houses from two to four I/O processors (IOPs), channel interfaces, and a shared memory section, called buffer memory. Figure 1-1 shows the IOS chassis.

IOS-C AND IOS-D DIFFERENCES

The IOS-C and IOS-D differ in the following ways:

- The IOS-C and IOS-D use different interrupt priority schemes.
- The IOS-C and IOS-D have different channel assignments and direct memory access (DMA) port assignments.
- The IOS-C interfaces a peripheral expander; the IOS-D interfaces a Versabus-Modular-Eurocard (VME) operator workstation (OWS).
- Each IOP in the IOS-C has a single 100-Mbyte channel for communication with CRI mainframe central memory; each IOP in the IOS-D has two 100-Mbyte channels for communication with CRI mainframe central memory.

Refer to Section 2 of this manual for information on interrupt priority schemes. Refer to Section 4 for information on central memory, peripheral expander, and operator workstation channel interfaces. Refer to Section 6 for information on channel assignments and DMA port assignments.

I/O PROCESSORS

Each IOP in the IOS is a fast, multipurpose computer capable of transferring data at 100 Mbytes/s. A 16-bit processor and fast bipolar local memory combine to support high-speed I/O operations. Each IOP has its own memory section, called local memory.

Figure 1-1. I/O Subsystem Chassis

The I/O capabilities make the IOPs useful for network control, mass storage access, and computer interfacing.

Each IOP has a control section, a computation section, an I/O section, and a local memory section. Section 2 of this manual describes each IOP section in detail.

The IOS configuration includes two, three, or four IOPs. The minimum configuration includes a master I/O processor (MIOP) and a buffer I/O processor (BIOP). An IOS can also include disk I/O processors (DIOPs) and auxiliary I/O processors (XIOPs). The IOS may contain two DIOPs, two XIOPs, or one BIOP and DIOP.

Figure 1-2 illustrates a four-processor IOS-C configuration. Figure 1-3 illustrates a four-processor IOS-D configuration.

The IOS uses an error multiplexer for detecting and reporting errors. This multiplexer passes channel error information and memory error information to a maintenance computer where the maintenance computer program logs the error information for later analysis.

I/O PROCESSOR FUNCTIONS

Each IOP in the IOS performs its functions independently. Software in each processor performs specific functions and is structured to perform these functions as efficiently as possible. Specific software functions depend on the IOS configuration and the peripheral equipment attached.

Each IOP logs information and keeps statistics about channel use and error detection and recovery. The IOPs use buffer memory to communicate with each other.

Each IOP has a 100-Mbyte/s channel for high-speed I/O transfers between buffer memory and central memory or buffer memory and a CRI SSD solid-state storage device. Refer to Figures 1-2 and 1-3.

The MIOP supports front-end interfaces and station software. THe MIOP has a 6-Mbyte/s control channel for communication with the CRI mainframe. The IOS-C MIOP controls I/O operations on a peripheral expander channel. The IOS-D MIOP provides an interface channel for a VME operator workstation. A deadstart operation begins with the MIOP. The DEADSTART switch (IOS-C) or the operator workstation (IOS-C or IOS-D) intitiates a deadstart operation on the MIOP. The MIOP initializes the contents of buffer memory and deadstarts the other IOPs. The MIOP, together with the BIOP, deadstarts the CRI mainframe.

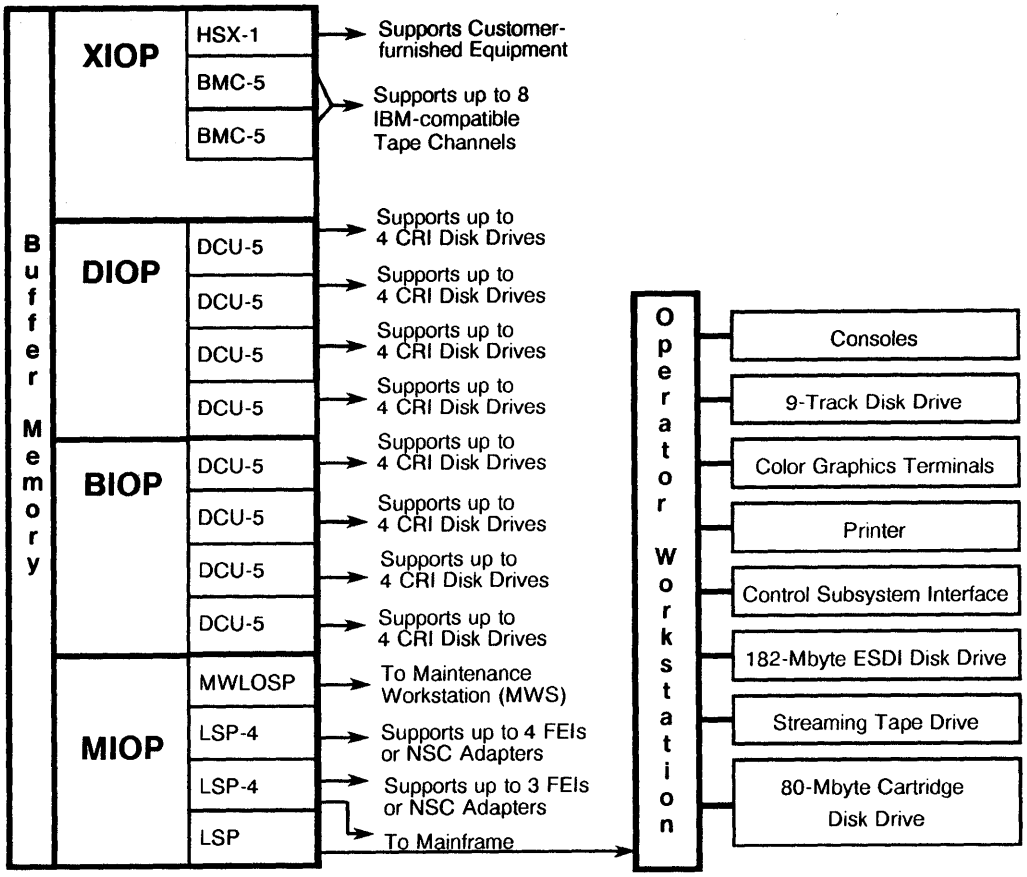


Figure 1-2. IOS-C Four-processor Configuration

The BIOP controls disk input and output to and from CRI disk storage units (DSUs) attached to its channels. Up to 16 DSUs attach to the BIOP on an IOS-C. Up to 12 DSUs attach to the BIOP on an IOS-D.

The DIOP also controls disk input and output to and from CRI DSUs attached to its channels. Up to 16 DSUs attach to the BIOP on an IOS-C. Up to 12 DSUs attach to the BIOP on an IOS-D.

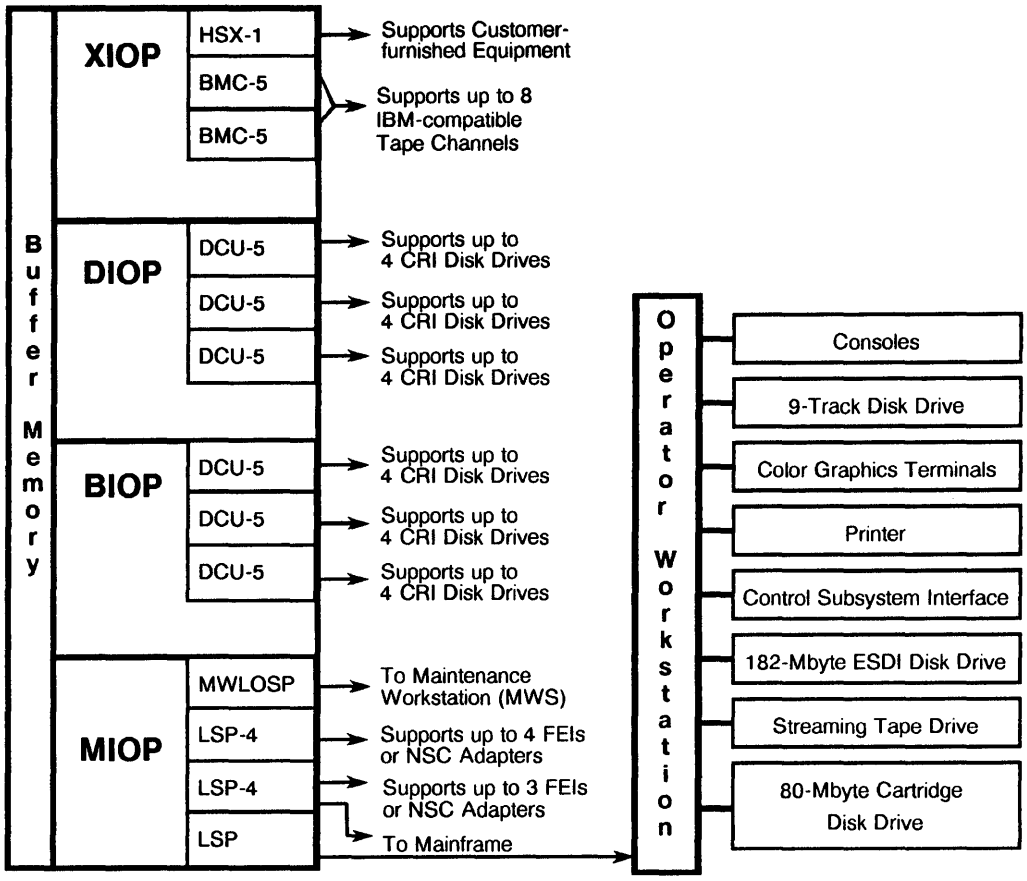


Figure 1-3. IOS-D Four-processor Configuration

The XIOP communicates with IBM-compatible equipment such as tape drives through block multiplexer controllers. The IOS-C XIOP controls up to 12 block multiplexer channels. The IOS-D XIOP controls up to 8 block multiplexer channels. The XIOP buffers data between the tape drives and central memory, and performs error-recovery procedures on errors detected while transferring data to or from tape.

Signals from the CRI mainframe control input and output for the MIOP, BIOP, DIOP, and XIOP.

I/O PROCESSOR CHANNEL INTERFACES

I/O processor channel interfaces adapt IOPs to various peripheral devices required to control the Cray computer system or provide data storage for the mainframe. Section 4 of this manual describes standard and optional interfaces used in the IOP.

I/O SUBSYSTEM BUFFER MEMORY

Buffer memory assists data transfers between peripheral devices and the CRI mainframe's central memory. Buffer memory is housed in the IOS chassis and stores 4 million, 8 million, 16 million, or 32 million 64-bit words. All IOPs share buffer memory, which uses single-error correction/ double-error detection (SECDED)† data protection.

Section 3 of this manual provides additional information on buffer memory.

I/O SUBSYSTEM CLOCK

A single clock controls the IOS. This clock is a crystal-controlled oscillator that runs at a frequency of 80 MHz with a clock period (CP) of 12.5 ns. The speed can be adjusted slightly for maintenance purposes. When operations require exact timing information, such as interval timing with the real-time clock, contact your CRI field engineer to verify the CP.

PERIPHERAL EXPANDER DEVICES

The IOS-C MIOP accesses devices through the peripheral expander. These devices include an 80-Mbyte disk drive, a magnetic tape unit, a printer/plotter, and an external clock.

These devices are not manufactured by CRI and are supported by original equipment manufacturer (OEM) documentation. This documentation is sent with the devices when they are shipped. Order additional copies or replacement copies of this documentation through the nearest representative of that company.

† The SECDED scheme is based on the error detection and correction method devised by R. W. Hamming.

2 - I/O PROCESSOR

Each I/O processor (IOP) consists of a control section, a computation section, an I/O section, and a local memory section. An IOP has a 12.5-ns clock period (CP) and runs 128 instruction codes as 16-bit (1-parcel) or 32-bit (2-parcel) instructions.

The IOP control section has an instruction stack, instruction control logic, a program exit stack, and control registers. The instruction stack holds 32 instructions from local memory. Small program loops run within the instruction stack without reference to local memory. Instruction control logic decodes instructions and transmits control signals to other sections of the IOP. The program exit stack stores return addresses for program subroutine calls or system hardware interrupts.

The IOP computation section contains operand registers, adder and shifter functional units, and an accumulator that work together to run program instructions. The 512 operand registers store data or memory addresses required to perform instructions. All operand registers are 16 bits wide. The adder and shifter functional units perform all arithmetic. Circuitry associated with the accumulator provides a logical product operation. The IOP accumulator is a 16-bit register within the computation section that temporarily stores operands or results. All data movement within the IOP uses the accumulator either as a source of data or as the destination for results. Transfers between memory and operand registers also use the accumulator.

The IOP I/O section consists of a series of interface channels that provide communication with peripheral devices, I/O subsystem (IOS) buffer memory, the CRI mainframe, and other IOPs. In an IOS model C (IOS-C), each IOP has a maximum of 42 interface channels. In an IOS model D (IOS-D), each IOP has a maximum of 44 interface channels. Refer to Section 6 in this manual for channel assignments of all interface channels.

Channels use busy and done flags to signal the IOPs and to transfer status information and function requests through the accumulator. External devices transfer data to and from the IOPs using one of the channel registers.

The IOP memory section is called local memory. It consists of four sections of four banks of random access memory (RAM). All sections function independently. Peripheral devices communicate with IOP local memory through six bidirectional direct memory access (DMA) ports. One or more channels are assigned to a single DMA port.

Local memory cycle time is 4 CPs for read operations, 4 CPs for accumulator write operations, and 6 CPs for I/O write operations. Access time (the time required to bring an operand from memory to the accumulator) is 7 CPs. Memory capacity is 65,536 16-bit parcels. A single-error correction, double-error detection (SECDED) network protects data transfered in and out of memory.

Figure 2-1 summarizes the IOP sections. Figure 2-2 is a block diagram of an IOP. The following subsections describe each part of an IOP in detail.

THE IOP CONTROL SECTION

The IOP control section controls the movement of instructions from local memory and decodes instructions into the appropriate function signals. The control section consists of an instruction stack, a program exit stack, control logic, and the following registers:

- Next instruction parcel (NIP) register
- Current instruction parcel (CIP) register
- B register
- Reference pointer (RP) register
- Destination pointer (DP) register
- Program address (P) register
- E register

The control section runs 128 instruction codes as 16-bit (1-parcel) or 32-bit (2-parcel) instructions. The instruction set includes branching and I/O instructions.

The program address counter controls the transfer of instructions from local memory to the instruction stack. Instructions issue from the instruction stack. The NIP and CIP registers decode the instructions into the appropriate control signals.

Instructions move data from a source to the accumulator and from the accumulator to a destination. Operand registers store operands and results. Functional units in the IOP computation section receive operand pairs and produce single results. The instruction designates one operand address, and the accumulator contains the other operand address. Typically, data flows from local memory to the accumulator, from the accumulator (with an operand) to a functional unit, back to the accumulator, and from the accumulator to local memory.

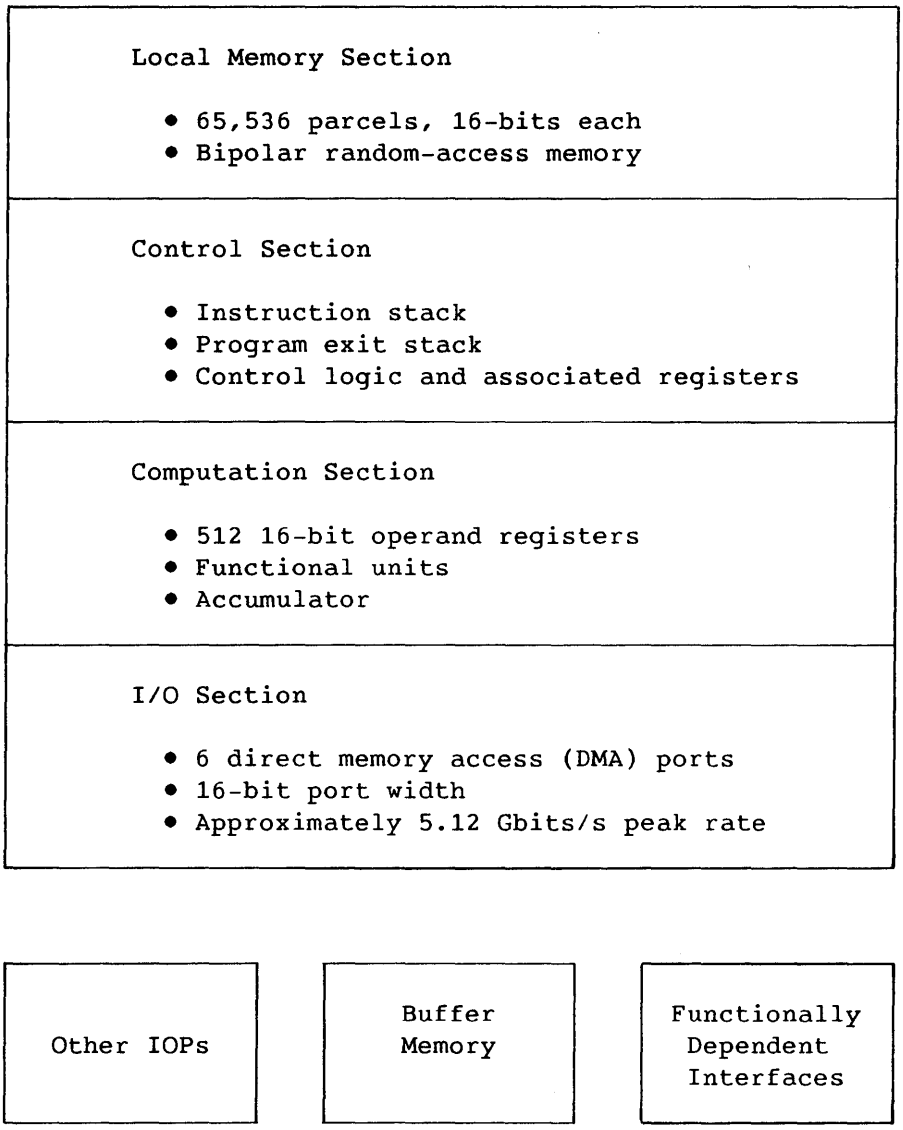


Figure 2-1. Basic Organization of an IOP

Instructions are 1 or 2 parcels in size. A 1-parcel instruction consists of a function code (*f*) field and a designator (*d*) field. A 2-parcel instruction consists of *f* and *d* fields and a constant (*k*) field (refer to Figure 2-3).

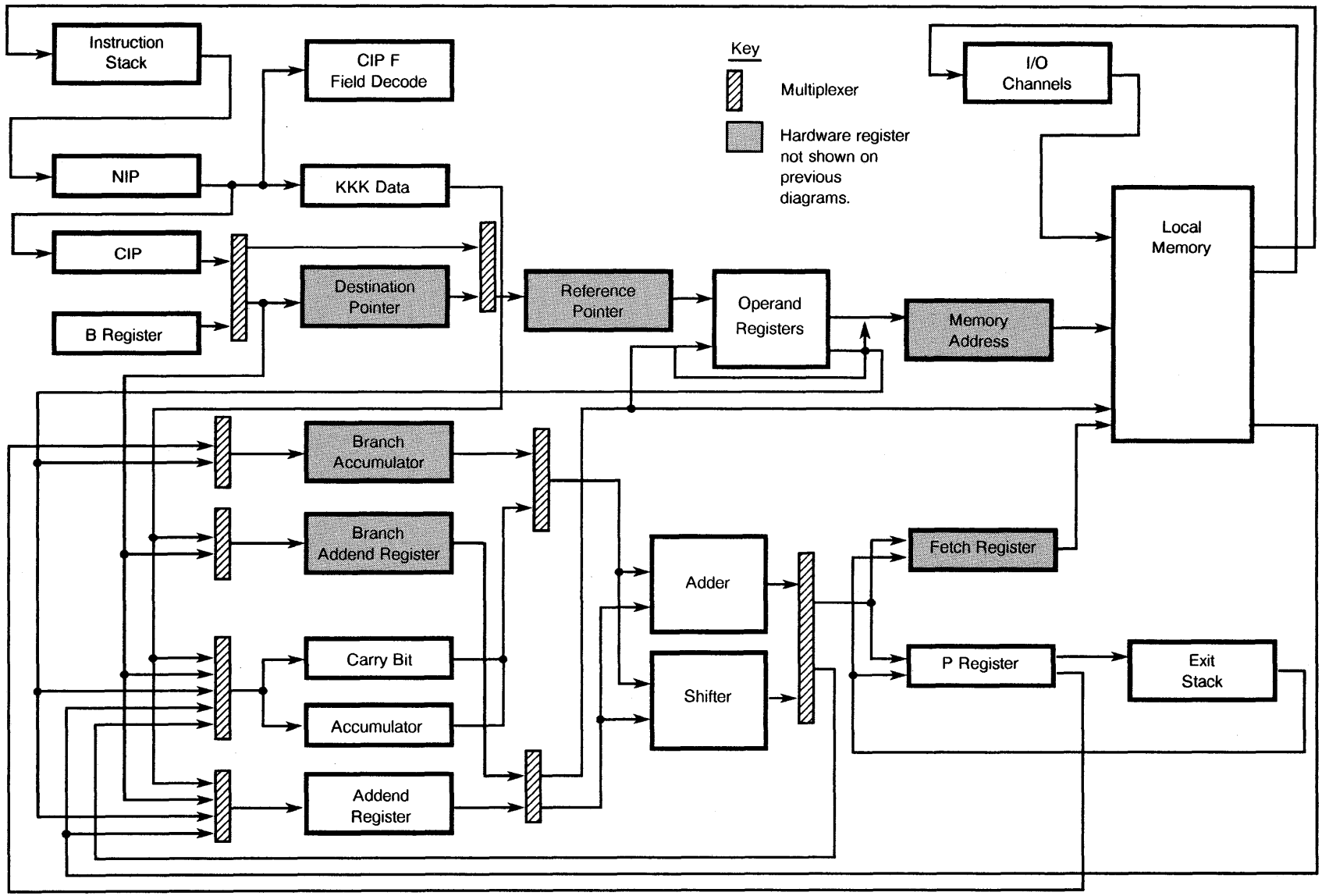


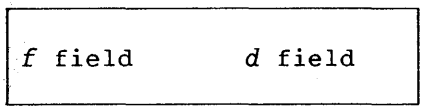
Figure 2-2. IOP Detailed Block Diagram

The *f* field is 7 bits in length and specifies the machine instruction to be issued. The d field contains one of the following values:

- An operand register
- An I/O channel number
- A displacement of data in a shift instruction
- A displacement forward or backward in program code for a branch instruction
- An operand

Refer to Section 7 of this manual for a detailed explanation of the instruction formats.

1-parcel instruction:



2-parcel instruction:

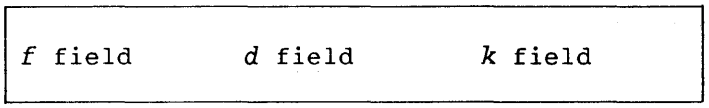


Figure 2-3. Instruction Formats

INSTRUCTION STACK

The instruction stack is a 32-parcel circular buffer that provides program branches with rapid access to program instructions. Instructions transfer from local memory to the instruction stack. Newly transferred instructions overwrite the longest-held parcels when the buffer is filled. Program branches take place either within the stack or outside of the stack. Short program loops run within the stack without reference to local memory.

The instruction stack contains 2 banks of registers with 16 parcels of program code in each bank (refer to Figure 2-4). Addresses alternate between the 2 banks of registers so that loading of data from storage interleaves with the readout of instructions.

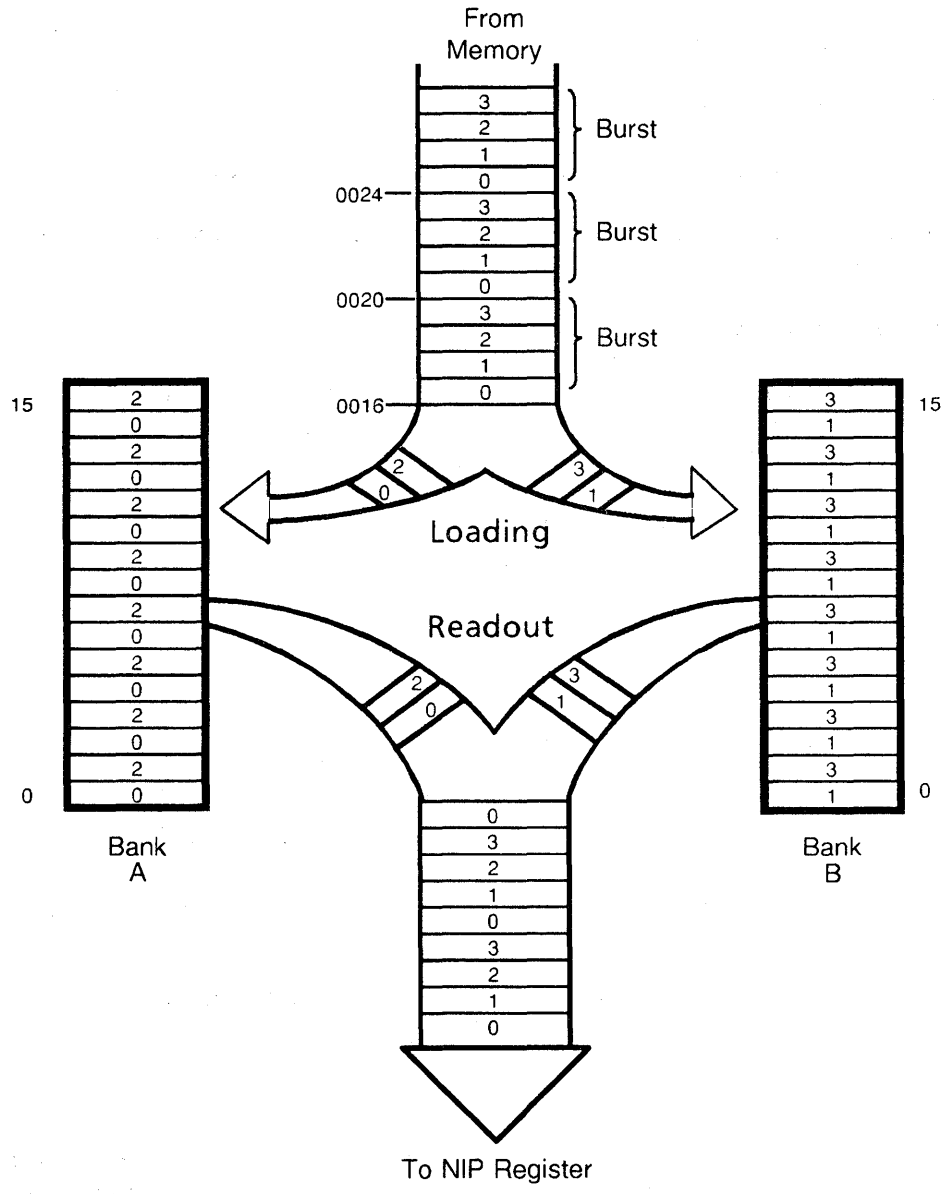


Figure 2-4. Instruction Stack Operation

Instructions transfer sequentially from local memory to the instruction stack in 4-parcel bursts. Each instruction transfer refers to a storage address that is a multiple of 4. The hardware loads the instruction stack sequentially at a rate of 1 parcel every CP if no local memory conflicts occur.

A series of instructions goes directly from local memory into the NIP if each instruction can load into the instruction stack and run in the same CP. The instructions move from local memory into the NIP register at the same time as they are loaded into the instruction stack. The bypass continues as long as the timing requirement is maintained or until issue-control circuitry sends a hold function request.

The instruction stack is a circular buffer. When the stack is full, it stores new code in location 0, overwriting the old code. The issue-control circuitry keeps a record of the segment of local memory currently in the instruction stack.

An internal fetch mechanism has access to instructions in the stack or enroute to it from local memory. Internal fetch operations are performed during instruction execution to ensure that the next instruction to be issued is in the stack.

Parity-checking circuits protect data transferred into and out of the instruction stack. IOP diagnostic modes check parity operation; refer to Section 8 of this manual for detailed information.

Instruction stack branching

Branch instructions create out-of-stack or in-stack branch conditions. Out-of-stack branch conditions cause local memory fetch operations. The fetch operations load the instruction stack beginning at location 0. After the instruction stack is loaded, the local memory and internal memory fetch mechanisms continue normal operation.

All absolute branch instructions (instructions that always cause a program branch: 074 through 077 and 120 through 137) create out-of-stack branch conditions. Relative branch instructions (070 through 073 and 100 through 117) can branch within the stack or out of the stack. The offset value in the d field and the locations of the instruction stack's load and stack pointers determine the in-stack or out-of-stack condition.

A forward relative branch instruction with an offset that is less than or equal to 11_8 always branches within the stack. A forward branch instruction with an offset greater than 11_8 may branch within the stack or out of the stack, depending on the location of the load and stack pointers.

A backward relative branch instruction with an offset that is less than or equal to 13_8 always branches within the stack. A backward relative branch instructions with an offset greater than 13_8 may branch within the stack or out of the stack, depending on the location of the load and stack pointers. If the instruction stack is being filled for the first time after an out-of-stack condition, a backward relative branch instruction is only valid to location 0 of the stack.

NEXT INSTRUCTION PARCEL REGISTER

The 16-bit next instruction parcel register receives the instruction parcel from the instruction stack (or local memory during bypass operation) and decodes it. An instruction parcel can remain in the NIP register more than 1 CP. The parcel transfers to the CIP register when the previous parcel leaves the CIP register.

CURRENT INSTRUCTION PARCEL REGISTER

The 16-bit current instruction parcel register receives the decoded instruction from the NIP register and holds the decoded instruction until all conditions for issue are met. The CIP register generates all control signals when the instruction issues. If the decoded *f* field shows the parcel to be the first of a 2-parcel instruction, the CIP register directs the NIP register to send the second parcel of the instruction to the addend register or accumulator.

B REGISTER

The 9-bit B register contains one of the following values, depending on the instruction:

- A pointer to one of 512 operand registers
- An I/O channel address for an I/O instruction
- An operand

REFERENCE POINTER REGISTER

The 9-bit reference pointer (RP) register addresses one of the 512 operand registers for reading or writing of data. The RP register receives its contents from the *d* field of the CIP register or from the B register.

DESTINATION POINTER REGISTER

The destination pointer (DP) register is sometimes the source of the operand register pointer. The DP register is used when two successive instructions use the same operand register. Compare logic checks the instructions to determine if both instructions use the same operand register.

The DP register receives its contents from the CIP register d field or from the B register when the instruction issues. When the register pointer is required, the DP register sends it to the RP register, using the same path as a register pointer from the CIP register.

Instruction issue is blocked for 1 CP when a transfer from the DP register to the RP register is required.

PROGRAM ADDRESS REGISTER

The 16-bit program address (P) register holds the local memory address of the instruction currently waiting to issue in in the CIP register. The address contained in the P register is always two program steps behind the instruction stack read-out address. The P register updates its contents automatically as each instruction issues.

PROGRAM EXIT STACK

The program exit stack (refer to Figure 2-5) is a set of 16 last-in, first-out registers that stores return addresses for program subroutine calls and program locations where loss of control occurred due to an I/O interrupt. The 4-bit E register addresses locations in the stack.

A program exit stack interrupt request issues when the stack empties or fills. The program reconfigures the stack by loading excess entries in local memory or retrieving entries from local memory. An unlimited number of subroutines can be nested by means of the program exit stack and the software.

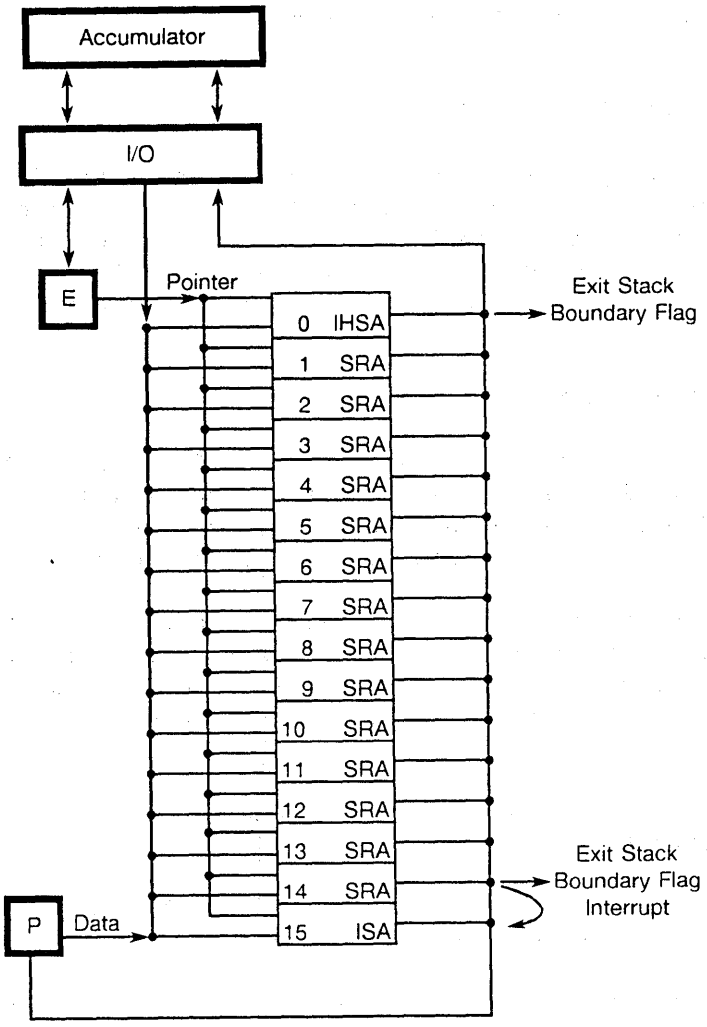
NOTE: Current software limits the use of the program exit stack to 13 nested subroutines.

I/O channel functions PXS : 0, 6, 7, 11, 13, and 16 access and modify the contents of the program exit stack and the E register.

NOTE: Allow at least 5 CPs after any modification of either the stack locations or the E pointer before performing any of the following:

- Performing a program exit operation
- Performing a return jump operation
- Enabling system interrupts
- Using data read from the program exit stack or the E pointer

Modify stack locations or the E pointer only when system interrupts are disabled.



ISHA Interrupt handler start address
SRA Subroutine return address or interrupted subroutine address (non-program exit stack interrupts)
ISA Interrupted subroutine address for the program exit stack interrupt only

Figure 2-5. Program Exit Stack

Parity circuits detect data errors in locations 1 through 15 of the program exit stack and set the program exit stack flag if an error occurs. The circuitry also flags an error if the program attempts to read out a location that is not written into. When servicing the program exit stack interrupt, check the current E pointer value. If the E pointer does not show that the stack is full or empty, the flag may indicate a parity error.

IOP diagnostics check program exit stack error-detection operation; refer to Section 8 of this manual for detailed information.

The deadstart program enters the interrupt handler starting address into position 0 of the stack. The starting address remains there throughout program operation. Interrupts cause the hardware to reference stack position 0 without reference to the E register.

Subroutine calls

Positions 1 through 14 in the program exit stack are available for storing subroutine return addresses.

A subroutine call from a routine or an interrupt advances the E pointer by 1 and stores the routine's return address in the stack. Upon exit from the subroutine, the program reads the return address from the current location in the stack and the E pointer decrements by 1 to point to the next highest stack location.

Reconfiguring the program exit stack

When the E pointer reaches 14 (the full limit of the stack), the program exit stack flag sets, causing an interrupt request and an interruption of the program. The interruption allows the software to reconfigure the stack by loading excess entries in local memory or by retrieving entries from local memory. The system and channel interrupts must be enabled.

NOTE: Current software limits the use of the program exit stack to 13 nested subroutines and does not perform the program exit stack interrupt request or the subsequent interruption of the program and interrupt service routine.

The following is the sequence for generating the program exit stack:

1. The E pointer goes to 14 because of a subroutine call.
2. The new return address loads to stack position 14.
3. The program exit stack flag is set, causing an interruption.
4. The next subroutine call enters a new value into the P register but no jump occurs.
5. The interrupt routine blocks issue of the next instruction.
6. The stack loads the interrupted address contained in the program address register to position 15.

7. The interrupt handler reconfigures the stack. Typically, The interrupt handler saves the higher half of the stack (positions 1 through 7) in local memory and moves the lower half of the stack (positions 8 through 15) to the higher part of the stack (positions 1 through 8). This procedure leaves the stack half full, allowing calls to deeper levels of subroutines or exits to higher levels.

8. The interrupt handler retrieves the address of the interrupted subroutine and exits to this subroutine.

When the E pointer reaches 0 and an exit instruction issues, the program exit stack boundary flag sets, causing an interrupt. The following sequence generates the interrupt with the E pointer at 0:

1. The E pointer reaches 0.

2. An exit instruction occurs.

3. The exit stack boundary flag sets, causing an interruption of the program.

4. The interrupt blocks issue of the next instruction.

5. The interrupt handler reconfigures the stack. Typically, the interrupt handler retrieves the next seven subroutine addresses from memory and sets E to 7. This procedure leaves the stack half full, allowing calls to deeper levels of subroutines or exits to higher levels.

6. The interrupt handler exits to the return address of the interrupted subroutine.

Program exit stack and I/O interrupts

The program exit stack treats an I/O interrupt much like a subroutine call. The monitor program stores the interrupted program address in the program exit stack at the next stack position and reads the interrupt routine entrance address from stack position 0. When servicing the interrupt, the hardware clears the system interrupt enable flag to disable further interrupts while the routine is in progress.

The interrupt handler routine sets the system interrupt enable flag after completion of the I/O operation. The exit at the end of the interrupt hander routine reads the return address for the interrupted program from the exit stack and returns control to the point in the program specified by the return address.

NOTE: Verify that enough levels are left available in the stack if return jumps are used in an interrupt handler. An interrupt with the exit stack pointer at 13 causes the pointer to go to 14. This condition leaves only one location open. If a return jump that causes a program fetch request (PFR) interrupt is issued with the stack pointer at 13, the return address goes to location 14 and the interrupt address goes to location 15. This condition leaves two interrupts present, the exit stack boundary and PFR. The PFR has the highest priority, but no stack locations are available. The stack pointer cannot increment from 15.

PROGRAM FETCH REQUEST FLAG

The program fetch request (PFR) flag provides a mechanism for calling the IOP monitor program when a new section of program code is required. The PFR flag sets during execution of jump instructions 074 through 077 and 120 through 137 when the instruction sequence finds a zero value in the operand register specified by *d*.

The PFR flag causes an interrupt request that discontinues the current program sequence at the completion of the interrupted instruction. The monitor program then reads the channel 1 interface input register contents. This register contains the operand register number specified by d. The monitor program uses the operand register number as an overlay to access a new section of program code. The monitor program finds the new section of code at the location equal to the old program location plus the operand register number d.

NOTE: The current operating system software does not use the mechanism provided by the program fetch request flag.

IOP COMPUTATION SECTION

The IOP computation section contains 512 operand registers, a memory address (MA) register, functional units, an accumulator, a carry bit register, and an addend register. These elements work together to run a program of instructions stored in memory.

An IOP adds, subtracts, left shifts, and right shifts operands using the adder and shifter functional units. The adder functional unit subtracts in two's-complement mode. The computation section does not perform floating-point arithmetic. The shifter functional unit performs left or right shifts of up to 31 bit positions in either circular or end-off shift mode. Circuitry associated with the accumulator provides a logical product operation.

All transfers to operand registers and all results from the functional units pass through the accumulator.

OPERAND REGISTERS

The 512 operand registers in an IOP act as storage locations for data, as index registers, and as indirect memory address (MA) registers. Each operand register contains 16 bits of data and has a 1-CP access time. The RP register addresses the operand registers for instruction decoding.

Data goes into the operand registers from the accumulator. Data leaving an operand register goes either to the accumulator or the addend register as operand data, or the MA register as memory address data. The IOP computation section references memory only through the operand registers.

MEMORY ADDRESS REGISTER

The 16-bit MA register holds the address for a read or write local memory reference. The MA register receives the address information from an operand register.

IOP FUNCTIONAL UNITS

The IOP computation section contains an adder functional unit and a shifter functional unit. The functional units perform all the arithmetic required by the instruction set.

Adder functional unit

The adder functional unit performs addition and subtraction. Subtraction is performed in two's-complement mode. Branch instructions use the adder functional unit for program address calculations.

Adder operands come from the instruction fields, the B register, the P register, the operand registers, and local memory. Operands come to the adder functional unit through the addend register and the accumulator. Operands from the accumulator are 17-bit operands; the high-order bit comes from the carry bit register. Operands from the addend register are 16-bit operands.

Adder functional unit results go to the accumulator, the P register, or the fetch register. Address calculation results for branch instructions go directly to the P register. All other adder results go to the accumulator for distribution. If the result has 17 bits, the high-order bit goes into the carry bit register.

Two's complement subtraction is performed in the adder functional unit by adding the one's complement of the addend register contents (the subtrahend) to the adder contents (the minuend) and then adding 1. The carry bit toggles if the result is negative or 0.

Both addition and subtraction require 1 CP; another CP is required to enter the results into the accumulator and carry bit register.

Shifter functional unit

The shifter functional unit shifts to the left or right up to 31 places. Shifts may be circular or end-off. Circular shifts cause the bits shifted out of the register to be shifted back into the opposite end of the register. End-off shifts cause the bits shifted out of the register to be discarded and the corresponding bits on the opposite end of the register to be filled with 0's. The shifter receives a 17-bit value to be shifted from the accumulator and from the carry bit register. The 5-bit shift count comes from the addend register. The shifter inverts the shift count for right shifts. The 17-bit results are returned to the accumulator and the carry bit register.

An operand cannot shift more than 31 places. If the shift count is 0, no shift occurs. If the count is greater than 16 for an end-off shift, the entire result is 0's. All shift operations treat the carry bit as the high-order bit (2^{16}) of the operand and the result.

Shift instructions incorporating left shifts, right shifts, circular shifts, and end-off shifts of any shift count all complete in 3 CPs.

IOP ACCUMULATOR

The 16-bit accumulator temporarily stores operands or results. All data movement within an IOP goes into or out of the accumulator. The instruction code specifies the source of data to the accumulator and the destination of data from the accumulator. The following IOP elements are sources and destinations for accumulator contents:

Sources
B register
Operand registers
Adder and shifter functional units
Local memory
I/O channels
CIP register *d* field
NIP register *k* field

Destinations
B register
Operand registers
Adder and shifter functional units
Local memory

The accumulator input logic performs logical product instructions. These instructions calculate the logical product of the accumulator contents and the input operand. The result goes to the accumulator. The logical product calculation requires no additional CPs.

Program branch instructions (070 through 137) form the destination address from two operands using a separate branch accumulator not visible to the programmer.

CARRY BIT REGISTER

The 1-bit, carry bit register holds the carry bit generated in the adder or shifter functional units. All addition, subtraction, and shift operations treat the carry bit as bit 2^{16} of the accumulator contents.

Instructions that test I/O channel flags set the carry bit. Some conditional jump and return jump instructions use the carry bit status as a criterion for the jump.

The carry bit register clears each time the accumulator loads.

ADDEND REGISTER

The 16-bit addend register supplies operands to the adder and shifter functional units. Whenever an instruction requires two operands, the accumulator supplies one operand and the addend register supplies the other. The addend register receives data from the B register, the NIP register, the operand registers, or local memory.

I/O PROCESSOR INPUT/OUTPUT SECTION

Each IOP supports up to 42 I/O channels on an IOS-C or 44 I/O channels on an IOS-D. The I/O channels are numbered in octal. The instruction or the B register contents select the channel number. Channels with standard functions always perform the same function, for example, channel 5 is the buffer memory channel on every MIOP, BIOP, DIOP, or XIOP. Sixteen channels and two ports have standard functions for each IOP on an IOS-C system. Eighteen channels and three ports have standard functions for each IOP on an IOS-D system. The remaining channels and DMA ports have variable functions, that is, they may be assigned uses according to the customer's needs.

All IOPs use two types of I/O channels: accumulator channels and DMA channels. Accumulator channels do not have direct access to local memory. DMA channels have direct access to local memory through DMA ports.

Up to four DMA I/O channels can be multiplexed onto a single DMA port. Each port can transfer a 16-bit parcel in either direction every CP. Data transfers at approximately 100 Mbytes/s per DMA port (maximum speed) in either direction. Input and output channels can be active simultaneously if they reference different memory sections.

All channels use busy and done flags to signal an IOP. All channels receive control information from the accumulator and send status information to the accumulator.

The following subsections describe accumulator channel control and data signals and DMA channel control and data signals. Section 4 of this manual describes individual channel interfaces.

INTERRUPT PRIORITIES

The interrupt priority scheme varies according to IOS model number. IOS-C channels are assigned interrupt priorities that cause them to be serviced from channel 0 through 51_8 in descending priority, with channel 0 having the highest priority and channel 51_8 the lowest priority. IOS-D channels are assigned interrupt priorities according to the following scheme:

1. Channels 0 through 17 (lower channel number = higher priority)
2. Channel 52
3. Channel 53
4. Channel 50
5. Channel 51
6. Channels 20 through 47 (lower channel number = higher priority)

I/O SPEEDS

The maximum speed of accumulator channels depends on the speed of the interrupt service routine.

A DMA port transfers data at the maximum rate of approximately 100 Mbytes/s in either direction.

ACCUMULATOR CHANNELS

Accumulator channels transfer data to and from the accumulator. Each accumulator channel interface has two 1-bit registers comprising the done and busy flags for the channel. The interface sets or clears these flags, and the IOP reads them using the Read Done and Read Busy control signals. The done flag generates an interrupt when interrupts are enabled.

Accumulator channel signals

Each accumulator channel uses the following signals to communicate with the channel interface of other devices:

- Function Designators signal - This signal contains the function code derived from bits 2^0 through 2^3 of the I/O instruction f field. The function code precedes all channel action and specifies different operations to different interfaces. Section 4 of this manual defines the function codes for each interface. The function code is stable on the signal lines for only 1 CP.

- Function Strobe signal - This signal indicates that the function code is present on the function designator lines.

- Accumulator Data signal - This signal contains data going to or from the IOP accumulator. The specific function code and channel interface determine the use of the data. Data leaving the accumulator is reliable only during the CP in which the Function Strobe signal is present.

- Read Done signal - This signal requests the condition of the done flag from the channel interface on the Busy/Done signal line.

- Read Busy signal - This signal requests the condition of the busy flag on the Busy/Done signal line.

- Busy/Done signal - This signal carries the response to the Read Done or Read Busy signal. The Busy/Done signal enters the IOP carry bit register 5 CPs after the Read Busy or Read Done signal issues.

- Master Clear signal - This signal goes to each channel interface only when the IOP is deadstarted. The function designators can invoke other clear functions.

- Clock signal - The IOP sends this signal to each channel interface to synchronize the logical operations. The clock signal period is approximately 12.5 ns. If the clock speed is varied for maintenance purposes, the pulse width stays constant and the period varies. This clock is independent of the clock used in the CRI mainframe.

- Interrupt signal - This signal causes an interrupt request in the IOP for the channel. An interruption occurs if an interrupt condition is present, if the channel interrupt enable flag is set for the channel, and if the system interrupt enable flag is set.

DMA CHANNELS

DMA channels perform high-speed block transfers. These channels have the same capabilities as accumulator channels but also allow direct access to local memory through the DMA ports.

DMA channel signals

A DMA channel uses all the signals described for accumulator channels and the following additional signals:

- Local Memory Data signals - These signals transfer data to and from local memory in groups of four 16-bit parcels, 1 parcel per CP in 4 consecutive CPs. A group of 16 lines carries data from local memory to the interface. A second group of 16 lines carries interface data to local memory.

  The interface sends the first parcel of write data in the CP following the Acknowledge Write signal from the IOP. The interface receives the first parcel of read data 7 CPs after the Acknowledge Read signal. Data is valid on the local memory data lines for only 1 CP.

- Local Memory Address signals - These signals come from the interface to the IOP on 14 lines. The interface sends the local memory address for the first parcel of a 4-parcel transfer. IOP logic generates the remaining addresses. The Local Memory Address signals must be stable when the interface sends the Request Read or the Request Write signals and remain stable until the interface receives the Acknowledge Read or Acknowledge Write signal.

- Request Read signal - The interface sends this signal to the IOP to request a local memory read operation. The interface sends the Request Read signal at the same time as it sends the the Local Memory Address signals.

- Request Write signal - The interface sends this signal to the IOP to request a local memory write operation. The interface sends the Request Write signal at the same time as it sends the Local Memory Address signals.

- Acknowledge Read signal - The IOP sends this signal when it is ready to perform a read operation. The interface sends the first data parcel of a 4-parcel group 7 CPs after it receives the Acknowledge Read signal.

- Acknowledge Write signal - The IOP sends this signal when it is ready to perform a write operation. The Acknowledge Write signal comes at least 1 CP after the Request Write signal; memory conflicts increase this time period. The channel interface sends the first parcel of data in the CP after it receives the Acknowledge Write signal.

DMA channel read sequence

Each channel read operation transfers 4 parcels from local memory to a peripheral device. The following is the read sequence:

1. An I/O instruction sends the local memory starting address to the channel interface.

2. An I/O instruction commands the interface to transfer data from local memory to a peripheral device.

3. The interface sends the Request Read and the Local Memory Address signals to the IOP.

4. The interface receives the Acknowledge Read signal from the IOP after a minimum delay of 1 CP.

5. The interface receives the first parcel 7 CPs after the Acknowledge Read signal and it receives the next 3 parcels in the next 3 CPs.

NOTE: Allow at least 2 CPs to elapse after the Acknowledge Read signal before sending the Request Read signal again.

DMA channel write sequence

Each DMA channel write operation transfers 4 parcels to local memory from a peripheral device. The following is the write sequence:

1. An I/O instruction sends the local memory starting address to the channel interface.

2. An I/O instruction commands the interface to transfer data from a peripheral device to local memory.

3. The interface sends the Request Write and Local Memory Address signals to the IOP.

4. The interface receives the Acknowledge Write signal from the IOP after a minimum delay of 1 CP.

5. The interface sends the first parcel of data to the IOP in the CP after it receives the Acknowledge Write signal. Each of the next 3 CPs transfers another parcel into local memory.

NOTE: Allow at least 2 CPs to elapse after the Acknowledge Write signal is received before raising the Request Write signal again.

CHANNEL INTERRUPT SEQUENCE

The monitor program interrupts an IOP program when an interrupt request is present and the system interrupt enable flag is set. If the interrupt request comes from an I/O channel, the corresponding channel interrupt enable flag must also be set.

The system interrupt enable flag enables interrupts for the entire IOP. Instruction 003 (I = 1) sets the system interrupt enable flag. The program enables interrupts after completion of the instruction immediately following the 003 instruction. Therefore, if an interrupt condition is present, the interruption occurs after the instruction immediately following the 003 instruction. Instruction 002 (I = 0) clears the system interrupt enable flag.

Hardware clears the system interrupt enable flag at the beginning of the interrupt sequence to prevent further interrupts. However, if a 003 instruction (I = 1, enable system interrupts) is pending when the program enters interrupt sequence, interrupts are reenabled during the interrupt sequence. To ensure that interrupts remain disabled during the interrupt sequence, the first instruction in the interrupt handler routine should always be 002 (I = 0, disable system interrupts).

Each channel has a channel interrupt enable flag that enables interrupts for that channel. The done flag interrupts the channel when interrupts are enabled. Channel function 7 sets the channel interrupt enable flag, and channel function 6 clears it. If an interrupt request is pending (channel done flag set) when a channel function 7 issues, the interruption occurs 3 CPs after the interrupt flag is sent. If an interrupt request is pending when a channel function 6 issues, the interrupt request is processed and the flag is cleared afterword.

The following is the interrupt sequence:

1. A 002 instruction clears the system interrupt enable flag and prevents further interrupts.

2. The monitor program stores the next address of the interrupted program in the exit stack.

3. The monitor program retrieves the entry address for the interrupt routine from the exit stack position 0.

4. The interrupt routine runs.

5. The monitor program sets the system interrupt enable flag and exits to the interrupted program when the interrupt routine finishes.

LOCAL MEMORY SECTION

Each IOP has a random-access, solid-state memory, called local memory. It consists of 4 sections of 4 banks each. Each 4-bank section functions independently.

Local memory has the following characteristics:

- Storage capacity is 65,536 16-bit parcels.
- Contains 16 banks of 4,096 parcels each.
- A read operation from local memory to the accumulator requires 7 CPs.
- One instruction may be fetched per CP when local memory is sequentially addressed.
- One data parcel may be fetched per CP when local memory is sequentially addressed.
- Each byte is protected by SECDED.
- Local memory has six full-duplex DMA ports
- Read-bank cycle time is 4 CPs.
- Write-bank cycle time is 6 CPs.

LOCAL MEMORY TIMING

Local memory cycle time is 4 CPs for a read operation and 6 CPs for a write operation. Access time is 7 CPs; access time is the time required to fetch an operand from local memory and send it to the accumulator.

Instructions transfer from local memory to the instruction stack at the rate of 1 parcel every CP in 4-parcel bursts. I/O operations transfer data from local memory at the same rate.

Local memory organization allows a memory reference every CP to sequential addresses. However, addresses in the same section can only be referenced once every 6 CPs by requests from the same DMA port. A single read or write operation transfers 1 parcel of operand data or 4 parcels of I/O data. Read and write operations to the peripheral expander transfer 1 parcel of I/O data per reference.

LOCAL MEMORY ORGANIZATION

Local memory is divided into 4 sections; each section has 4 banks. Each bank contains 4,096 16-bit parcels.

Each 4-bank section of local memory has separate access paths, sequence controls, write data registers, and read data registers. The 4 banks within a section share a common sequence control; reference to any 1 bank of the section also references the other 3 banks in the section, but only data from the addressed bank transfers to the requester.

An operand reference transfers a single parcel of data; an I/O reference transfers 4 parcels. The local memory section is free to begin another reference 4 CPs after initiation of the previous I/O reference.

An I/O write operation can assemble data for a section of data in the write data registers while an I/O read operation is reading data from the same section of local memory.

LOCAL MEMORY ACCESS

Each section of local memory has three 16-bit data paths for reading data and two 16-bit data paths for writing data. One read path and one write path go to the accumulator. One read path and one write path go to the IOP's I/O section to make up a DMA port. The last read path goes to the instruction stack and carries instruction parcels.

LOCAL MEMORY ADDRESSING

A local memory address contains 16 bits (refer to Figure 2-6). The low-order 4 bits specify the section and bank of the section to be addressed. The high-order 12 bits specify an address within the storage chip.

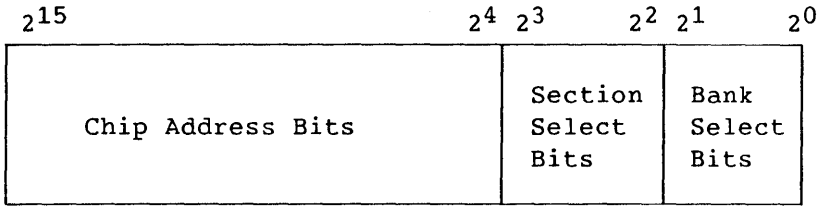


Figure 2-6. Local Memory Address Format

LOCAL MEMORY ERROR CORRECTION

A local memory error-correction and detection network ensures that all data written into memory or read from memory is correct. The network checks each byte using SECDED. If 1 bit of a byte is in error, the SECDED network corrects the error before transferring the byte. If 2 bits of the same byte are in error, the SECDED network detects and reports the error. A double-bit error interrupts the IOP if interrupts are enabled. If 3 or more bits are in error, results are unpredictable.

SECDED operation

The SECDED circuitry adds a 5-bit check byte to every data byte during a write operation. Each check bit is an even parity bit for a specific group of data bits. Figure 2-7 shows the designated bits of the data word that are assigned to each parity bit in the check byte. An X in the horizontal row indicates that the data bit contributes to the generation of the corresponding check bit. Figure 2-8 gives an example of check byte generation.

Check bits are stored in memory at the same location as the data byte. When the data byte is read from memory, SECDED circuitry generates a new set of parity bits, using the matrix shown in Figure 2-7. These new bits are even parity bits of the data byte and the old check bits. The resulting 5 parity bits are called syndrome (S) bits, shown as bits 8 through 12 in Figure 2-7.

	Check Byte					Bit in Data Byte							
	12	11	10	9	8	7	6	5	4	3	2	1	0
S0					X	X		X		X		X	X
S1				X		X	X			X	X		X
S2			X			X	X	X	X				
S3		X						X	X	X	X	X	
S4	X						X		X		X	X	X
Single-error Syndrome Code	20	10	4	2	1	7	26	15	34	13	32	31	23

Figure 2-7. Check Byte Matrix

Data Byte = 152_8 (01101010_2)

	Check Byte	Bit in Data Byte
S0	1	0 1 1 1 0
S1	0	0 1 1 0 0
S2	0	0 1 1 0
S3	1	1 0 1 0 1
S4	0	1 0 0 1 0
		0 1 1 0 1 0 1 0

Check Byte = 11_8 (01001_2)

Figure 2-8. Check Byte Generation Example

The SECDED matrix decodes the syndrome bits and determines the error condition by following these rules:

- If all syndrome bits are 0, no error has occurred.
- If only 1 syndrome bit is 1, the associated check bit is in error.
- If more than 1 syndrome bit is 1, and the parity of all syndrome bits S0 through S4 is even, then a double error occurred within the data bits or check bits.
- If more than 1 syndrome bit is 1, and the parity of all syndrome bits is odd, then a single and correctable error occurred. SECDED circuitry decodes the syndrome bits to identify the bit in error.

An error in 3 or more data bits causes unpredictable results.

Local memory SECDED logic uses the same error-correction scheme used in the CRI mainframe's central memory. Refer to one of the mainframe reference manuals listed in the preface for further information on SECDED operation.

Buffer memory is a random-access, solid-state memory shared by all I/O processors (IOPs) in an I/O subsystem (IOS) chassis. Buffer memory holds data for transfer between peripheral devices (such as disks and tapes) and central memory or a CRI SSD solid-state storage device.

Buffer memory capacity is 4, 8, 16, or 32 million 64-bit words. Data is refreshed every 4 ms. The refresh operation does not affect the random access capability, although it can cause bank conflicts.

This section describes buffer memory speed, organization, access, addressing, and error protection.

BUFFER MEMORY SPEED

Buffer memory cycle time is 26 clock periods (CPs). Cycle time is the time required between storage location references.

BUFFER MEMORY ORGANIZATION

Buffer memory is organized into groups, sections, and banks. A 4-Mword buffer memory has 1 group; an 8-Mword buffer memory has 2 groups. Each group has 2 sections, and each section has 4 banks.

Each address in buffer memory contains 64 data bits and 8 check bits. Data transfers to and from buffer memory in 16-bit parcels. Four parcels make up one 64-bit word. Buffer memory receives and sends in a 0, 1, 2, 3 sequence; Figure 3-1 shows the format.

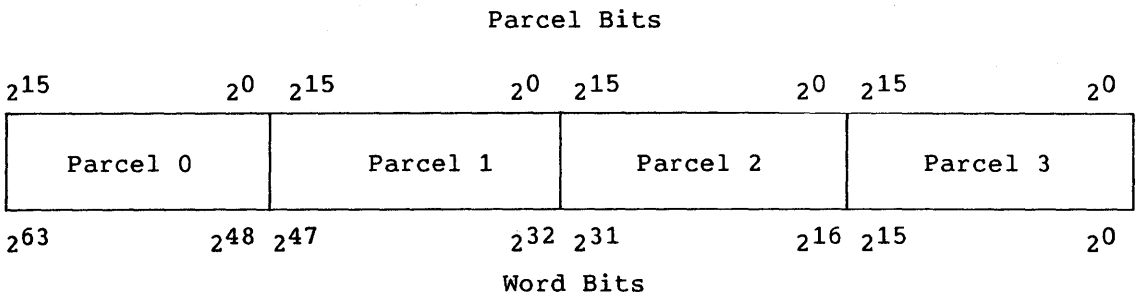


Figure 3-1. Buffer Memory Word Format

BUFFER MEMORY ACCESS

Buffer memory has four ports, each capable of connecting to an IOP, as shown in Figure 3-2. A channel interface adapts one local memory DMA port to a buffer memory port. Each of the four local memory ports has access to all banks through a time-sharing scanner. If a port requests a reference to a bank that is busy, the scanner makes a reservation for that port. The port gains access to the bank as soon as it becomes available.

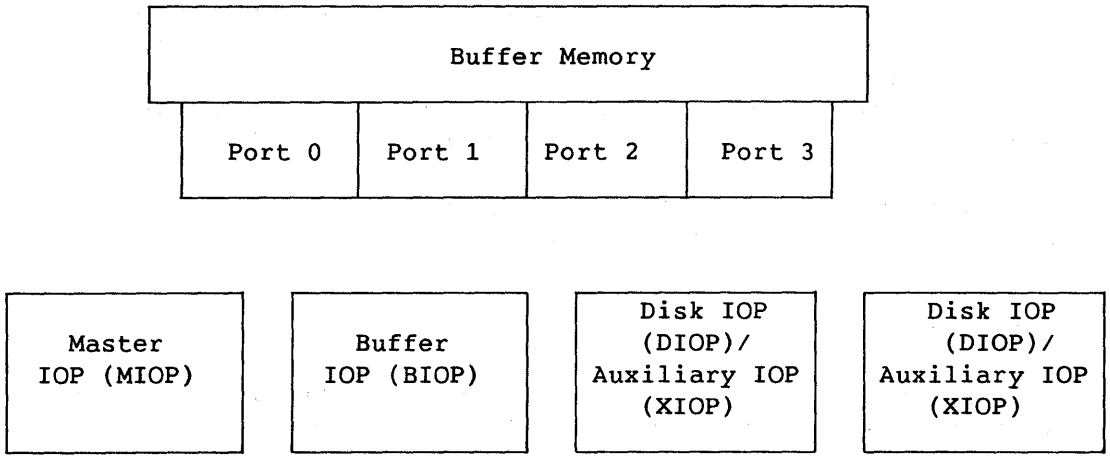


Figure 3-2. Buffer Memory Port Assignments

Buffer memory access time is 12 CPs. However, communication speed depends on buffer memory size, the number of ports attempting to use local memory, and the number of IOPs attempting to use buffer memory.

BUFFER MEMORY ADDRESSING

A 23-bit address identifies each 64-bit word. The IOP accumulator supplies the address through a channel interface using the MOS : 2 and MOS : 3 functions as shown in Figure 3-3. The address represents a physical location in buffer memory.

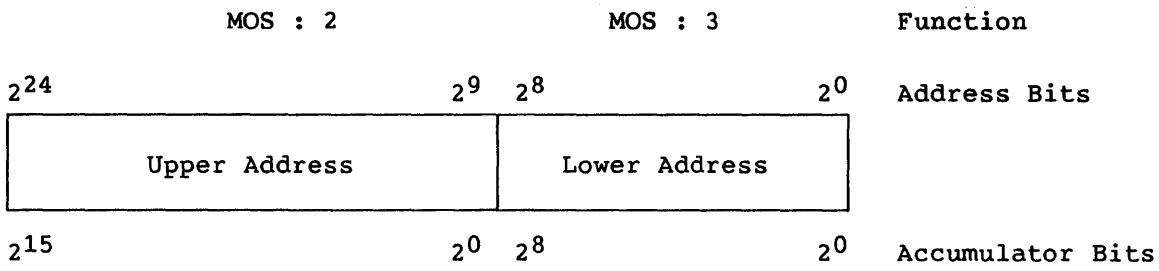


Figure 3-3. Buffer Memory Address Formation

BUFFER MEMORY ERROR PROTECTION

Buffer memory uses single-error correction/double-error detection (SECDED) logic to determine if the data has been altered by the storage cycle. Refer to "Local Memory Error Correction" in Section 2 of this manual for a detailed description of SECDED operation.

4 - I/O PROCESSOR CHANNEL INTERFACES

Channel interfaces adapt an I/O processor (IOP) to other devices. A channel interface buffers data, generates control signals for the attached device, and multiplexes several attached devices into the same IOP channel. This section describes the channel interfaces and the function codes that control them.

CHANNEL INTERFACE CHARACTERISTICS

Each IOP interfaces 42 I/O channels on the I/O subsystem model C (IOS-C) or 44 I/O channels on the I/O subsystem model D (IOS-D). Output data is transfered from an IOP accumulator to a channel interface register. Input data is transfered from an interface register to the accumulator. Direct memory access (DMA) ports carry data for block transfers into or out of local memory.

A series of function codes, generated by instructions 140 through 177 control data transfers and channel interface actions. The d designator in the program instruction or the B register contents address the channel. An channel interface interprets up to 16 function codes. The function codes are interpreted differently by different channel interfaces. Table 4-1 lists specific IOP channel functions. The following functions are common to all channel interfaces, except the I/O request interface and the peripheral expander interface:

Function Number	Description
0	Clears the busy and done flags and places the channel in an idle status.
6	Clears the channel interrupt flag, blocking any further interrupt requests from the channel.
7	Sets the channel interrupt enable flag and enables interrupt requests from the channel.

Most interfaces have a busy flag and a done flag, which the interface sets and clears during function execution. The busy flag is normally set when the channel is active and cleared when the channel is idle. However, busy and done flag activity varies with the specific channel interface and the specific function code.

Table 4-1. IOP Channel Functions and Descriptions

Channel Name	Function	Description
I/O request	IOR : 10	Reads the interrupt channel number
Program fetch request	PFR : 0	Clears the program fetch request flag
	PFR : 6	Disables channel interrupts
	PFR : 7	Enables channel interrupts
	PFR : 10	Reads the operand register number
Program exit stack	PXS : 0	Clears the exit stack boundary flag
	PXS : 6	Disables channel interrupts
	PXS : 7	Enables channel interrupts
	PXS : 10	Reads the exit stack pointer
	PXS : 11	Reads the exit stack address
	PXS : 13	Reads the history log
	PXS : 14	Enters the exit stack pointer
	PXS : 15	Enters the exit stack address
	PXS : 16	Enters diagnostic mode
Machine hardware error	MHE : 0	Clears the local memory parity error flag
	MHE : 6	Disables channel interrupts
	MHE : 7	Enables channel interrupts
Real-time clock	RTC : 0	Clears the done flag
	RTC : 6	Disables channel interrupts
	RTC : 7	Enables channel interrupts
	RTC : 10	Reads the real-time clock
Buffer memory	MOS : 0	Clears the busy and done flags
	MOS : 1	Enters the local memory address
	MOS : 2	Enters the buffer memory address upper bits
	MOS : 3	Enters the buffer memory address lower bits
	MOS : 4	Reads buffer memory; enters the block length
	MOS : 5	Writes buffer memory; enters the block length
	MOS : 6	Disables channel interrupts
	MOS : 7	Enables channel interrupts
	MOS : 10	Reads the bypass modes; Reads the error bits
	MOS : 14	Sets the control register flags
	MOS : 15	Sets the second control register flags
	MOS : 16	Enters bypass modes

Table 4-1. IOP Channel Functions and Descriptions (continued)

Channel Name	Function	Description
IOP input	AIA : 0	Clears the done flag
	AIA : 6	Disables channel interrupts
	AIA : 7	Enables channel interrupts
	AIA : 10	Reads the accumulator input and resume
IOP output	AOA : 0	Clears the busy and done flags
	AOA : 1	Enters control bits from the accumulator
	AOA : 6	Disables channel interrupts
	AOA : 7	Enables channel interrupts
	AOA : 14	Sets busy flag; output accumulator data
Central memory or SSD input	HIA : 0	Clears the busy and done flags
	HIA : 1	Enters the local memory address
	HIA : 2	Enters the upper central memory address
	HIA : 3	Enters the lower central memory address
	HIA : 4	Enters the block length; starts a transfer
	HIA : 6	Disables channel interrupts
	HIA : 7	Enables channel interrupts
	HIA : 10	Reads the syndrome code or error code
	HIA : 14	Enters diagnostic mode
Central memory or SSD output	HOA : 0	Clears the busy and done flags
	HOA : 1	Enters the local memory address
	HOA : 2	Enters the upper central memory address
	HOA : 3	Enters the lower central memory address
	HOA : 5	Enters the block length; starts a transfer
	HOA : 6	Disables channel interrupts
	HOA : 7	Enables channel interrupts
	HOA : 10	Reads the error code
	HOA : 14	Enters diagnostic mode
Mainframe/ maintenance input	LIA : 0	Clears the busy and done flags
	LIA : 1	Enters the local memory address; starts input
	LIA : 2	Enters the parcel count for transfer
	LIA : 3	Clears the channel parity error flags
	LIA : 4	Clears the ready waiting flag
	LIA : 6	Disables channel interrupts
	LIA : 7	Enables channel interrupts
	LIA : 10	Reads the local memory address
	LIA : 11	Reads status (ready waiting, parity error)
Mainframe output	LOA : 0	Clears the busy and done flags
	LOA : 1	Enters the local memory address, starts output

Table 4-1. IOP Channel Functions and Descriptions (continued)

Channel Name	Function	Description
Mainframe output (continued)	LOA : 2	Enters the parcel count
	LOA : 3	Clears the error flag
	LOA : 4	Sets and clears external control signals
	LOA : 6	Disables channel interrupts
	LOA : 7	Enables channel interrupts
	LOA : 10	Reads the local memory address
	LOA : 11	Reads the processor number and error flag
Console keyboard (Accumulator channel)	TIA : 0	Clears the done flag
	TIA : 3	Sets the baud rate
	TIA : 6	Disables channel interrupts
	TIA : 7	Enables channel interrupts
	TIA : 10	Reads data
Console display (Accumulator channel)	TOA : 0	Clears the busy and done flags
	TOA : 6	Disables channel interrupts
	TOA : 7	Enables channel interrupts
	TOA : 14	Sends accumulator data to the display
IOS-C Peripheral Expander (Accumulator channel)	EXB : 0	Idles the channel
	EXB : 1	Requests the A input register contents
	EXB : 2	Requests the B input register contents
	EXB : 3	Requests the C input register contents
	EXB : 4	Reads the busy/done flag and interrupt address
	EXB : 5	Loads the device address
	EXB : 6	Sends the interface mask
	EXB : 7	Sets the interrupt mode
	EXB : 10	Reads the data bus status
	EXB : 11	Reads status 1
	EXB : 13	Reads status 2
	EXB : 14	Sends data to the A output register
	EXB : 15	Sends data to the B output register
	EXB : 16	Sends data to the C output register
	EXB : 17	Sends control signals
High-speed external communications (HSX) input (DMA channel)	XIA : 0	Clears the busy and done flags
	XIA : 1	Enters the local memory address
	XIA : 3	Enters special functions
	XIA : 4	Enters the block length
	XIA : 6	Disables channel interrupts
	XIA : 6	Enables channel interrupts
	XIA : 10	Reads the local memory address

Table 4-1. IOP Channel Functions and Descriptions (continued)

Channel Name	Function	Description
High speed external communications (HSX) output (DMA channel)	XOA : 0	Clears busy and done flags and error conditons
	XOA : 1	Enters the local memory address
	XOA : 3	Enters special functions
	XOA : 5	Enters the block length; transfers the block
	XOA : 6	Disables channel interrupts
	XOA : 7	Enables channel interrupts
	XOA : 10	Reads the local memory address
	XOA : 11	Reads the output status
Front-end/ operator work-station/IOS-D peripheral expander input (DMA channel)	CIA : 0	Clears the busy and done flags
	CIA : 1	Enters local memory address; starts input
	CIA : 2	Enters the parcel count
	CIA : 3	Clears the channel parity error flags
	CIA : 4	Clears the data waiting flag
	CIA : 6	Disables channel interrupts
	CIA : 7	Enables channel interrupts
	CIA : 10	Reads the local memory address
	CIA : 11	Reads status (data waiting, parity errors)
Front-end/ operator work-station output (DMA channel)	COA : 0	Clears the busy and done flags
	COA : 1	Enters the local memory address
	COA : 2	Enters the parcel count
	COA : 3	Clears the error flag
	COA : 4	Sets and clears external control signals
	COA : 6	Disables channel interrupts
	COA : 7	Enables channel interrupts
	COA : 10	Reads the local memory address
	COA : 11	Reads the sequence error status
Block multiplexer channel (DMA channel)	BMA : 0	Clears channel control
	BMA : 1	Sends reset functions
	BMA : 2	Sends commands to the control units
	BMA : 3	Reads the request-in address
	BMA : 4	Performs single byte I/O
	BMA : 5	Performs delay counter diagnostics
	BMA : 6	Disables channel interrupts
	BMA : 7	Enables channel interrupts
	BMA : 10	Reads the local memory address
	BMA : 11	Reads the byte count
	BMA : 12	Reads status
	BMA : 13	Reads the input tags

Table 4-1. IOP Channel Functions and Descriptions (continued)

Channel Name	Function	Description
	BMA : 14	Enters the local memory address
	BMA : 15	Enters the byte count
	BMA : 16	Enters the device address
	BMA : 17	Enters the output tags
DCU-4 disk storage unit (DMA channel)	DKA : 0	Clears the busy and done flags
	DKA : 1	Selects a mode or requests status
	DKA : 2	Reads data from the disk
	DKA : 3	Writes data to the disk
	DKA : 4	Selects a head group
	DKA : 5	Selects a cylinder
	DKA : 6	Disables channel interrupts
	DKA : 7	Enables channel interrupts
	DKA : 10	Reads the local memory address
	DKA : 11	Reads the status response
	DKA : 14	Enters the local memory address
	DKA : 15	Tests the status response register
DCU-5 disk storage unit (DMA channel)	DIA : 0	Clears the busy, done, and error flags
	DIA : 1	Selects a control or status operation
	DIA : 2	Reads data from the disk
	DIA : 3	Writes data to the disk
	DIA : 4	Performs a diagnostic echo
	DIA : 5	Selects a cylinder
	DIA : 6	Disables channel interrupts
	DIA : 7	Enables channel interrupts
	DIA : 10	Reads local memory address register 0
	DIA : 11	Reads local memory address register 1
	DIA : 12	Reads status register 0
	DIA : 13	Reads status register 1
	DIA : 14	Enters local memory address register 0
	DIA : 15	Enters local memory address register 1
	DIA : 16	Enters the next read or write parameter
	DIA : 17	Selects a diagnostic mode

The IOP program detects the busy flag status by performing instructions 041 and 043.

The done flag normally signals the IOP program that channel activity has reached a point requiring program action. The channel interface hardware normally controls the done flag, but the program can also set or clear the done flag. Instructions 040 and 042 detect the state of the done flag.

The channel interface generates an interrupt when the done flag sets, if the channel interrupt enable and system interrupt enable flags are also set. Use an IOR : 10 function to detect channel interrupt conditions when the system interrupt enable flag is clear.

CHANNEL TIMING CONSIDERATIONS

The following timing considerations apply to channel functions for all interfaces:

- Channel functions 6 and 7 (clear and set channel interrupts) become effective 3 clock periods (CPs) after issue.
- Allow 1 CP after issuing a channel function 6 or 7 before checking the interrupt channel number with an IOR : 10 function.
- Allow 1 CP after issuing any function before checking the busy or done flags.

Section 6 of this manual lists each channel function according to channel assignment. The following pages describe each function in detail.

STANDARD IOP CHANNELS

Each IOP contains interface logic for 24 standard channels in the IOS-C or 26 on the IOS-D. Table 4-2 describes channel assignments. Channels 006 through 013_8 connect the system IOPs and therefore vary for different IOPs.

ERROR LOGGING

The IOPs use an error-multiplexing system for detecting and reporting errors. This system detects single and multiple errors, even if a multiple error is embedded in a burst of single-bit errors. All IOPs pass channel error information and memory error information to a maintenance computer. The maintenance computer program logs the error information for later analysis.

The error-multiplexing system multiplexes the following channels from a single IOP:

- The buffer memory channel
- Up to four local memory channels
- Two central memory channel pairs (input and output)

If necessary, additional multiplex modules provide access to the error log for up to ten central memory channel pairs.

Table 4-2. IOP Standard Channel Assignments

Channel (octal)	Assignment
000	Interrupt request
001	Program fetch request
002	Program exit stack
003	Machine hardware error
004	Real-time clock
005	Buffer memory interface
006	IOP-to-IOP input
007	IOP-to-IOP output
010	IOP-to-IOP input
011	IOP-to-IOP output
012	IOP-to-IOP input
013	IOP-to-IOP output
014	Central memory input
015	Central memory output
040	Console keyboard
041	Console display
042	Console keyboard
043	Console display
044	Console keyboard
045	Console display
046	Console keyboard
047	Console display
050	Mainframe/maintenance input
051	Mainframe/maintenance output
052	Central memory input (IOS-D only)
053	Central memory output (IOS-D only)

I/O REQUEST CHANNEL

The I/O request channel (channel 0) reads interrupt requests. The channel 0 done flag is always set, and the channel 0 busy flag is always cleared.

The interface register for this channel contains the number of the highest priority channel currently requesting an interrupt. Clearing the interrupt enable flag or the done flag for the corresponding channel changes the contents of the register.

Specify channel 0 with instructions 040 through 043 to set or clear the carry bit. These instructions force the carry bit to the condition of the busy or done flags.

The only I/O request channel function is IOR : 10 (read interrupt channel number). This function loads accumulator bits 2^0 through 2^9 with the number of the highest priority channel currently requesting an interrupt and forces accumulator bits 2^{10} through 2^{15} to 0. If no channel interrupts are pending, this function loads all 0's into the accumulator.

PROGRAM FETCH REQUEST CHANNEL

The program fetch request channel (channel 1) and the program fetch request flag provide a mechanism for calling the IOP monitor program when a new section of program code is required. This channel treats the done flag as the program fetch request flag. The flag sets during execution of jump instructions 074 through 077 and 120 through 137 when the instruction sequence finds a zero value in operand register *d*. This channel does not have a busy flag.

A 9-bit register in the channel interface holds the operand register number. This register is cleared and a new register number is entered at the time the program fetch request flag is set. Refer to the "Program Fetch Request Flag" subsection in Section 2 of this manual for more information on the program fetch request channel.

NOTE: The current operating system software does not use the program fetch request feature.

The following are functions for the program fetch request channel:

Function	Description
PFR : 0	Clears the program fetch request flag. The interface treats the program fetch request flag as if it was the done flag.

Function	Description
PFR : 6	Clears the channel interrupt enable flag. This function does not affect the program fetch request flag.
PFR : 7	Sets the channel interrupt enable flag. This function does not affect the program fetch request flag.
PFR : 10	Replaces the contents of the accumulator with the contents of the channel's interface register. This function loads the interface register contents into accumulator bits 2^0 through 2^8 and forces accumulator bits 2^9 through 2^{15} to 0. The interface register contents do not change until a new PFR interrupt request occurs. This function clears the program fetch request flag after the function is complete.

PROGRAM EXIT STACK CHANNEL

The program exit stack channel (channel 2) controls the program exit stack hardware and provides the monitor program with information needed to reconfigure the stack. Refer to Section 2 of this manual for a description of program exit stack operation.

This channel has a program exit stack flag in place of a done flag. The channnel has no busy flag. The following are functions for the program exit stack:

Function	Description
PXS : 0	Clears the program exit stack flag. This function also clears the program exit stack diagnostic mode and the history log read-out mode. Refer to Section 8 for more information on diagnostic modes.
PXS : 6	Clears the channel interrupt enable flag. This function does not change the program exit stack flag.
PXS : 7	Sets the channel interrupt enable flag. This function does not change the program exit stack flag.
PXS : 10	Loads the E designator contents into accumulator bits 2^0 through 2^3 and forces the high-order accumulator bits to 0.
PXS : 11	Replaces the accumulator contents with the contents of the program exit stack address currently specified by the E designator.

Function	Description
PXS : 13	Reads the back contents of the jump history log. When this function issues, the program enters a jump history log mode. Return jumps and interrupts are not allowed until a PXS : 0 function clears the jump history log mode status. Refer to the description of diagnostic channels 404 through 407 in Section 8 of this manual for more information on the jump history log.
PXS : 14	Replaces the E designator contents with accumulator bits 2^0 through 2^3. When issuing this function, allow 5 CPs for enabling system interrupts or issuing return jumps or exit instructions. Issue a PXS : 14 function only when system interrupts are disabled.
PXS : 15	Enters the accumulator contents into the program exit stack at the address currently specified by the E designator. When issuing this function, allow 5 CPs for enabling system interrupts or issuing return jumps or exit instructions. Use this function only when system interrupts are disabled.
PXS : 16	Enters program exit stack diagnostic mode. If accumulator bit 2^0 is set, this mode forces parity bits to 0 on a write operation. If accumulator bit 2^1 is set, this mode forces parity bits to 1 on a write operation. A PXS : 0 function or a second PXS : 16 function with with all accumulator bits equal to 0 terminates the diagnostic mode.

NOTE: The exit stack is both an I/O device and an integral part of the IOP. Return and exit instructions and interrupts use the exit stack values immediately. Access by the I/O channel to the exit stack takes 4 CPs to complete. Allow time for the I/O channel to complete the transfer before use of the exit stack value by a return or exit instruction, or an interrupt.

****

CAUTION

At least 5 CPs must elapse before using any data read from the program exit stack. Any attempt to use a value changed in the stack within 5 CPs after it changes can scramble the program sequence.

****

MACHINE HARDWARE ERROR CHANNEL

The machine hardware error channel (channel 3) connects to the local memory single-error correction/double-error detection (SECDED) circuits. When the interface receives a report of an uncorrectable error (double-bit error) in local memory, the error flag sets and an external error-logging device records the error.

The following are functions for the machine hardware error channel:

Function	Description
MHE : 0	Clears the error flag.
MHE : 6	Clears the channel interrupt enable flag.
MHE : 7	Sets the channel interrupt enable flag.

REAL-TIME CLOCK CHANNEL

The real-time clock (RTC) channel (channel 4) connects to the IOS's RTC. The RTC is a 17-bit counter/timer that interrupts the IOP at 1-ms intervals and increments every CP. Upon reaching a count of 234177_8, the RTC clears to 0, sets the RTC channel done flag, and continues incrementing. The RTC channel does not have a busy flag.

The RTC cannot be set. To time an interval, the program reads the clock at the beginning of the interval, stores the value in a register, and reads the clock at the end of the interval. The above interval-timing sequence adds 8 CPs to the measured interval. The following algorithm determines the actual timed interval if no RTC interrupts occur:

$$\text{Time (ns)}_8 = (\text{RTC ending}_8 - \text{RTC beginning}_8 - 4) \times 2 \text{ (CP ns)}$$

If an RTC interrupt occurs during the interval, the algorithm is as follows:

$$\text{Time (ns)}_8 = [(\text{RTC ending}_8 - \text{RTC beginning}_8 - 4) + (116077_8 \times \text{number of interrupts})] \times 2 \text{ (CP ns)}$$

To eliminate the possibility of an RTC interrupt during a time interval of less than 1 ms, synchronize the beginning of the interval with an RTC interrupt.

The following are functions for the RTC clock channel:

Function	Description
RTC : 0	Clears the done flag.

Function	Description
RTC : 6	Clears the channel interrupt enable flag.
RTC : 7	Sets the channel interrupt enable flag.
RTC : 10	Reads the RTC count into the accumulator. Because the accumulator only holds 16 bits, the low-order bit of the RTC counter is ignored, providing a timing accuracy of 2 CPs, and a count of 116077_8.

BUFFER MEMORY INTERFACE CHANNEL

The buffer memory channel (channel 5) allows the IOP program to transfer blocks of data between local memory and buffer memory in either direction.

The buffer memory channel together with the central memory channel form a bypass channel that transfers data directly between buffer memory and central memory. Refer to Section 5 of this manual for information on the bypass operation.

The buffer memory channel has the following interface registers to control block copy (non-bypass) operations:

- A 24-bit buffer memory address register. An MOS : 2 function and an MOS : 3 function load the address from the accumulator. The hardware forces 2 lower-order bits of the register contents to 0, leaving a value that is a multiple of 4.
- A 14-bit local memory address register. An MOS : 1 function loads the address from the accumulator.
- A 14-bit block length register. MOS : 4 and MOS : 5 functions, which initiate read and write operations, also load the block length (in buffer memory words) from the accumulator. A value of 0 in the block length register indicates a maximum block length of 65,536 parcels.

The contents of all three registers are 0 at the end of a block copy.

The busy flag remains set at the end of a transfer if one of the following conditions occur:

- A multiple memory error on a buffer memory read operation
- An uncorrectable data error on a buffer memory read or write operation
- A central memory channel error with the bypass mode set
- A forced function error caused by an MOS : 0 function issued to buffer memory with bypass mode set

The following are functions for the buffer memory channel:

Function	Description
MOS : 0	Clears the busy and done flags, if the bypass mode is not active. If the bypass mode is active, this function forces a function error to the central memory channel interface. The central memory channel interface function error terminates the bypass mode and forces an error to the buffer memory channel.
MOS : 1	Enters the accumulator contents into the local memory address register and forces the 2 low-order bits to 0.
MOS : 2	Enters the 15 low-order bits of the accumulator as the 15 high-order bits of the buffer memory address. Refer to Figure 4-1.
MOS : 3	Enters the 9 low-order bits of the accumulator as the 9 low-order bits of the buffer memory address. Refer to Figure 4-1.

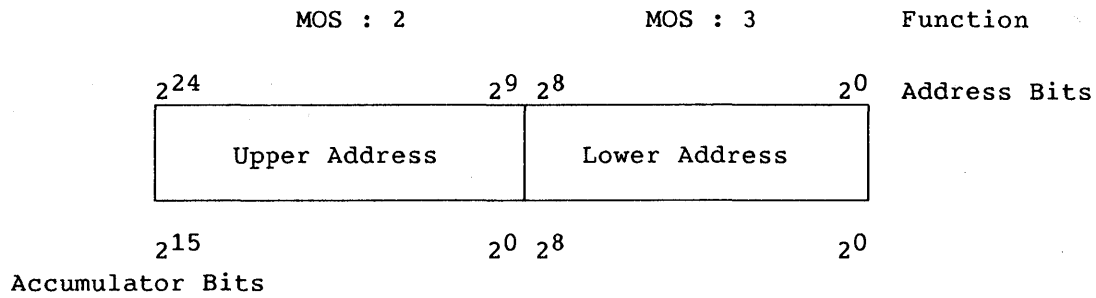


Figure 4-1. Buffer Memory Address Formation

MOS : 4	Initiates a transfer of the data block from buffer memory. Initially, the busy flag sets and the done flag clears. The function enters the 14 low-order accumulator bits into the block length register before performing the transfer. Upon completion of the transfer, the done flag sets and the busy flag clears. The busy flag remains set if a multiple-bit error occurred in the transfer. SECDED circuitry automatically corrects single-bit errors.
MOS : 5	Initiates the transfer of a data block to buffer memory. Initially, the busy flag sets and the done flag clears. The function enters the 14 low-order accumulator bits

Function	Description

into the block length register before performing the transfer. Upon completion of the transfer, the done flag sets and the busy flag clears.

MOS : 6 Clears the channel interrupt enable flag.

MOS : 7 Sets the channel interrupt enable flag.

MOS : 10 Reads the bypass mode bits if accumulator bit 2^1=1; reads the error bits if accumulator bit 2^0=1.

The function samples accumulator bits 2^0 and 2^1, clears the accumulator, and enters bypass or error information into the accumulator, depending on the previous state of bits 2^0 and 2^1. If bit 2^0 of the accumulator is set when the function issues, the function enters the following error information into the accumulator:

Bit	Decription
2^0	When set, a double-bit error occurred on a buffer memory read operation.
2^1	When set, a data error occurred from the channel to buffer memory.
2^2	When set, a data error occurred from buffer memory to the channel.
2^3	When set, a central memory channel error occurred and the last transfer was in bypass mode.

If bit 2^1 of the accumulator is set when the function issues, the function enters the following bypass information into the accumulator:

Bit	Decription
2^0	The channel is currently in bypass mode and linked with the buffer memory input channel.
2^1	The channel is currently in bypass mode and linked with the buffer memory output channel.
2^2	The previous transfer was a bypass operation for the central memory input channel.

Function	Description

MOS : 13 (continued)

Bit	Decription
2^3	The previous transfer was a bypass operation for the central memory output channel.

MOS : 14 Enters the accumulator contents (bit positions 2^1 and 2^2) into a diagnostic control register. Bit 2^1 disables write check bits when set. Bit 2^2 disables the memory refresh operation when set. Clearing bit 2^1 or 2^2 enables a normal write check bit or memory refresh operation. This function is used for diagnostic purposes only and is restricted to diagnostic mode.

MOS : 15 Enters the accumulator bits 2^1, 2^2, and 2^3 into a second diagnostic control register. This function is used for diagnostic purposes only and is restricted to diagnostic mode. The bits are described in the following list:

Bit	Description
2^1	Disables local memory/buffer memory channel parity generation (even parity causes 1-parity bits; odd parity causes 0-parity bits).
2^2	Enables maintenance mode.
2^3	Disables SECDED.

MOS : 16 Enters the bypass mode. If accumulator bit positions 2^0 through 2^3 contain a value of 14_8, buffer memory forms a link with the central memory input channel (channel 14). If accumulator bit positions 2^0 through 2^3 contain a value of 15_8, buffer memory forms a link with the central memory output channel (channel 15).

Error conditions

When an error occurs on the buffer memory channel, the interface sets the busy and done flags. An MOS : 0 function clears the busy and done flags before another read or write operation is initiated.

Interface deadstart

An IOP uses the buffer memory channel interface when performing a deadstart operation through the IOP-to-IOP output channel. The IOP-to-IOP output channel interface initiates the deadstart operation by

setting and then clearing the Deadstart and Master Clear signals. These signals initiate a transfer of a 64,000 parcel block of data from buffer memory to the IOP. The completed transfer interrupts the IOP.

Interface dead dump

An IOP uses the buffer memory channel interface when performing a dead dump through the IOP-to-IOP output channel. The IOP-to-IOP output channel interface initiates the dead dump by setting and clearing the Dead Dump and Master Clear signals. These signals initiate a transfer of a 64,000 parcel block of data from the IOP local memory to buffer memory. The completed transfer does not interrupt the IOP.

IOP-TO-IOP INPUT CHANNELS

An IOP has three input channels (channels 6, 10, and 12) for communicating with other IOPs. The input channels have no busy flags. Each channel interface uses a single 16-bit register to temporarily store data parcels being transferred. The register clears when the data enters the receiving IOP's accumulator.

The following functions are for the IOP-to-IOP input channels:

Function	Description
AIA : 0	Clears the done flag. Allow 1 CP before checking the done flag.
AIA : 6	Clears the channel interrupt enable flag. Allow 1 CP before checking the interrupt channel number (function IOR : 10).
AIA : 7	Sets the channel interrupt enable flag. Allow 1 CP before checking the interrupt channel number (function IOR : 10).
AIA : 10	Reads the accumulator data of the sending IOP into the accumulator of the IOP into the IOP issuing the function. The issuing IOP maintains control of the channel. The data does not remain in the interface register after it has been read. This function generates an interrupt on the corresponding output channel in the sending IOP.

IOP-TO-IOP OUTPUT CHANNELS

An IOP has three output channels (channels 7, 11, and 13) for communicating with other IOPs. These channels can master clear, deadstart, or dead dump the other IOPs in the same IOS.

Each output channel has a 16-bit data register and a 4-bit control register. The data register holds transmitted data until the data is loaded into the receiving IOP's accumulator. The data register clears after the data transfer. The control register receives the 4 low-order bits of the sending IOP's accumulator after an AOA : 4 function. The control register bits are described in the following list:

Bit	Description
2^0	Master clear
2^1	Dead start
2^2	Dead dump/master clear buffer memory
2^3	Short transfer

A master clear operation of the receiving IOP occurs when control register bit 2^0 sets and remains set at least 200 ns. The master clear interrupts the program in the receiving IOP.

A dead-start operation from buffer memory occurs when control register bits 2^0 and 2^1 set simultaneously and remain set at least 200 ns. When the bits clear, a 65,000 parcel block transfers from buffer memory (starting at address 0) to the receiving IOP local memory (starting at address 0). The transfer is complete in approximately 2 ms and interrupts the IOP.

A short dead-start operation from buffer memory occurs when control register bits 2^0, 2^1, and 2^3 set simultaneously and remain set at least 200 ns. When the bits clear, a 4096-parcel transfer begins. The short transfer is complete in approximately 100 ms and interrupts the IOP.

A dead-dump operation from local memory to buffer memory occurs when bits 2^0 and 2^2 set simultaneously and remain set at least 200 ns. When both bits clear, a 65,000 parcel block transfers from local memory (starting at address 0) to buffer memory (starting at address 0). The dead dump is complete in approximately 2 ms and does not interrupt the IOP on completion.

A short dead dump from local memory to buffer memory occurs when bits 2^0, 2^2, and 2^3 set simultaneously and remain set for at least 200 ns. When all bits clear, a 4096-parcel transfer begins. The transfer is complete in approximately 100 ms and does not interrupt the IOP on completion.

A master clearing of buffer memory occurs when control register bit 2^2 sets while bit 2^0 is clear. This operation aborts any current buffer memory transfers and resynchronizes the error-logging channel with buffer memory. A master clearing of buffer memory is normally restricted to diagnostic mode.

****

CAUTION

Do not master clear buffer memory during a data transfer. This disrupts the transfer and can cause data loss.

****

The following are functions for the IOP-to-IOP output channels:

Function	Description
AOA : 0	Clears the busy and done flags.
AOA : 1	Enters the 4 low-order accumulator bits into the control register.
AOA : 6	Clears the channel interrupt enable flag.
AOA : 7	Sets the channel interrupt enable flag.
AOA : 14	Sets the busy flag and sends data from the accumulator to the receiving IOP. When the receiving IOP accepts the data, the busy flag clears and the done flag sets. This function generates an interrupt on the corresponding input channel in the receiving IOP. Allow 1 CP before checking the busy or done flags.

CENTRAL MEMORY AND SSD INPUT CHANNEL

The central memory input channels (channel 14 on an IOP in an IOS-C chassis; channels 14 and 52 on an IOP in an IOS-D chassis) transfer data to IOP local memory from mainframe central memory or an SSD. The channels carry 64 bits of data in parallel with 8 check bits, and all transfers take place in 16-word bursts. A separate 16-bit channel carries address information.

NOTE: References to central memory in the remainder of the central memory input channel subsection also apply to the SSD, unless otherwise noted.

The central memory channel input signal sequence is more complex than an ordinary DMA channel. The central memory channel passes more control signals and more data protection information than an ordinary DMA channel. Figure 4-2 illustrates the signal exchange between an IOP and central memory or an SSD.

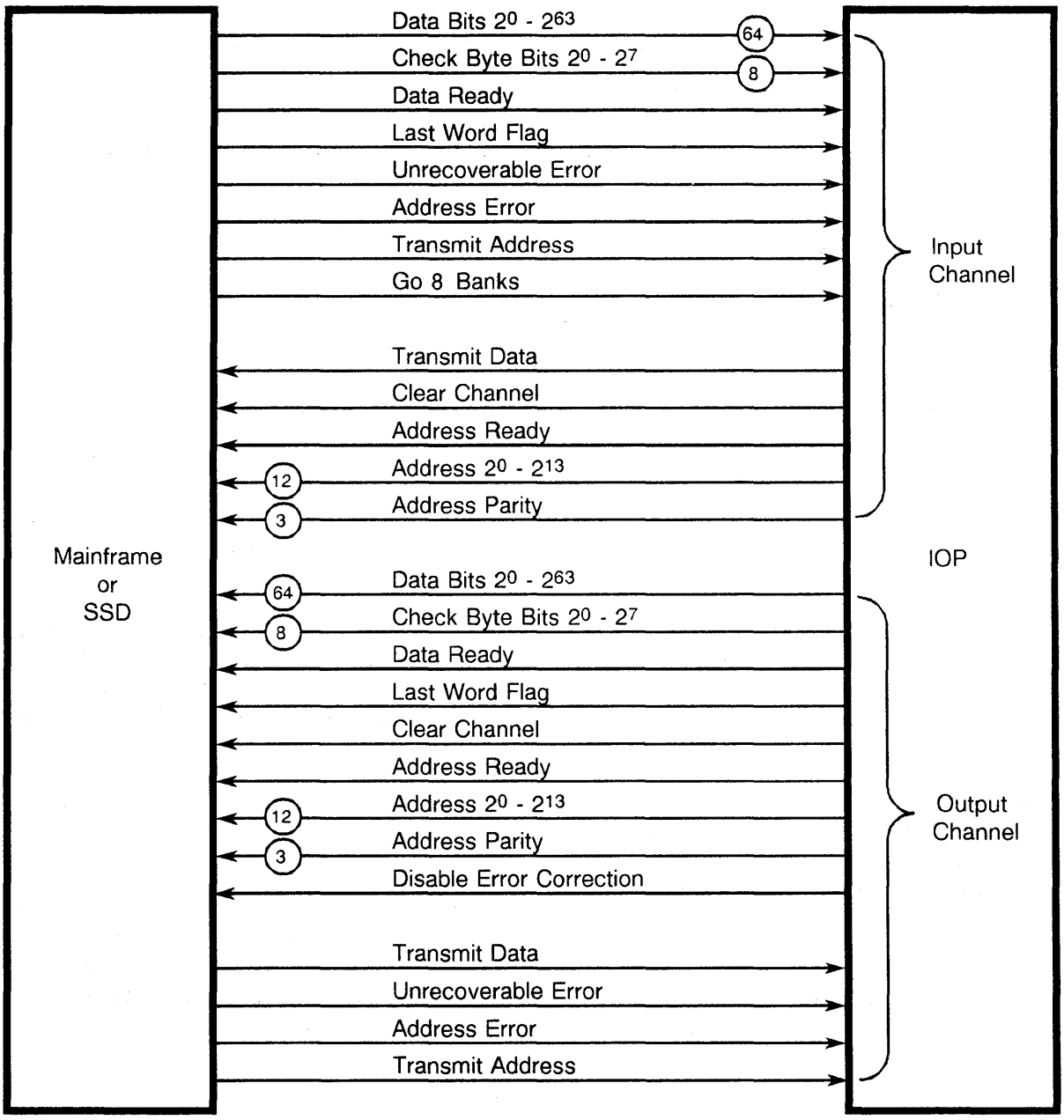


Figure 4-2. Central Memory Channel Signals

The following paragraphs describe central memory input channel signals, sequences, and functions. Section 5 of this manual describes central memory channel operation for bypass mode.

Central Memory and SSD Input Channel signals

The following paragraphs describe the central memory channel input signals (from central memory or an SSD to local memory) and the signal timing. Table 4-3 shows the sequence of signals for transferring data from central memory to an IOP. The following are central memory input channel signals:

- Data signals - Data transfers in 64-bit words over 64 lines. Data is valid on the data lines 2 CPs after the leading edge of the Data Ready signal and stays valid for 6 CPs.

- Check-byte signals - The Check-byte signals consist of 8 error-check bits sent by central memory to the IOP to verify data accuracy. The Check-byte signals are valid 2 CPs after the leading edge of the Check-byte signals and stay valid for 6 CPs.

- Data Ready signal - The Data Ready signal indicates data on the channel is valid. The leading edge of the Data Ready signal precedes the data signals by 2 CPs, and and the Data Ready signal is valid for 1 CP (2 CPs if the signal is from an SSD). The minimum period for a Data Ready signal is 6 CPs.

- Last Word signal - The Last Word signal indicates the most recent data word of the transfer is on the data lines, and the central memory word count is 0. This signal has the same timing and duration as the Data signal.

- Unrecoverable Error signal - The Unrecoverable Error signal indicates that an unrecoverable error occurred in the transfer at central memory. The channel becomes inactive and remains so until a Clear Channel signal is sent to central memory from the IOP. Unrecoverable errors include the following conditions:

    - An address parity error.

    - A Transmit Data signal received by central memory when less than three Address Ready signals were received.

    - More than three Address Ready signals received by central memory.

    - An uncorrectable-data error from central memory.

  An Unrecoverable Error signal continues until the IOP sends a Clear Channel signal.

Table 4-3. Central Memory Input Channel Sequence

Step	IOP Action	Mainframe or SSD Action
1.	Activates the channel.	Performs no action.
2.	Raises the Enable Block signal.	Performs no action.
3.	Raises the Enable Burst signal.	Performs no action.
4.	Stores the first burst in I/O memory.	Transfers the first burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer.
5.	Raises the Enable Burst signal.	Performs no action.
6.	Stores the words in I/O memory.	Transfers the second burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer.
7.	Raises the Enable Burst signal.	Performs no action.
8.	Stores the final burst in I/O memory.	Transfers the final burst of data in 64-bit words. Raises the Data Clock signal before each 64-bit transfer. Raises the Block End signal.
9.	Drops the Enable Block signal.	Performs no action.

- Address Error signal - The Address Error signal indicates an unrecoverable address error occurred (one of the first three conditions listed under Unrecoverable Error signal). The IOP samples the Address Error signal on the leading edge of the Unrecoverable Error signal.

- Transmit Address signal - The Transmit Address signal indicates the central memory side of the interface is inactive and ready to receive an address from the IOP. The central memory side of the interface clears the Transmit Address signal after receiving three address ready signals and does not set the Transmit Address signal again until the transfer is complete. The Transmit Address signal also clears if there is an error condition on the central memory side of the channel and does not set again until a Clear Channel signal is sent.

- Go 8 Banks signal - The Go 8 Banks signal indicates that data is transferred in 8-word blocks (8-bank mode). When the channel is connected to an SSD, the signal is always clear.

- Transmit Data signal - The Transmit Data signal indicates that the IOP is ready to accept a block of data from central memory. The IOP clears the Transmit Data signal when the channel is inactive. The signal sets at least 8 CPs after the trailing edge of the third Address Ready signal.

  The Transmit Data signal remains set until the IOP receives the first Data Ready signal for that data block. During transmission, the IOP asserts the Transmit Data signal whenever the IOP channel's input data buffer can receive 16 words without overflowing.

  Central memory samples the Transmit Data signal at the beginning of each data block. If the Transmit Data signal is set, central memory sends the 16-word (or 8-word) data block. If the Transmit Data signal is clear, no data is transmitted until the Transmit Data signal sets.

  NOTE: Transfers vary according on the device connected to the central memory input channel:

  - All data transfers from CRAY X-MP computer systems are in 16-word blocks, with the exception of the last block, which can be an odd length.

- The size of the first block from a CRAY-1 S computer system is manipulated to adjust the subsequent blocks to 16-bank central memory boundaries. The number of data words sent in the first block equals 16 minus the number specified by bits 2^0 and 2^1 of the central memory starting address. The next block begins with section 0. If the Go 8 Banks signal is set, the number of data words in the first block is 8 minus the address 2^0 value.

- The size of the first block from an SSD is manipulated to adjust the subsequent blocks to 16-bank central memory boundaries. The number of data words sent in the first block equals 16 minus the number specified by bits 2^6 and 2^7 of the SSD starting address. The next block begins with section 0. The SSD keeps the Go 8 Banks signal clear.

- Clear-channel signal - The Clear-channel signal clears all error conditions in the channel and remains set a minimum of 8 CPs. The Address Ready signal follows the trailing edge of the Clear-channel signal by at least 8 CPs.

- Address Ready signal - The Address Ready signal precedes the address information. The leading edge of the Address Ready signal leads the address information by 2 CPs and stays set for 2 CPs. The minimum period for Address Ready signals is 6 CPs, leading edge to leading edge. Three Address Ready signals initiate a transfer.

  The first Address Ready signal accompanies bits 2^{10} through 2^{23} of the beginning central memory address or bits 2^{16} through 2^{29} of the beginning SSD address on the address lines. The second Address Ready signal accompanies bits 2^0 through 2^9 of the beginning central memory address or bits 2^6 through 2^{15} of the beginning SSD address, and bits 2^{12} through 2^{13} of the transfer word count. The third Address Ready signal to central memory accompanies bits 2^0 through 2^{11} of the transfer word count (for an SSD, bits 2^0 through 2^5 of the transfer word count must always be 0).

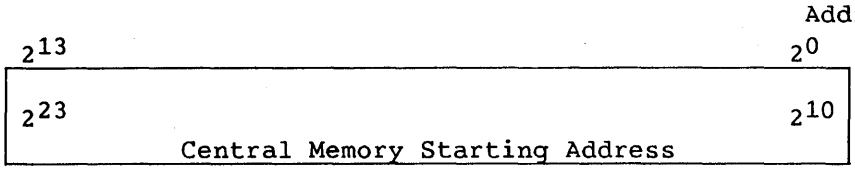
- Address signals - The 16 address lines carry the central memory starting address and the transfer word count in three address transfers. Figures 4-3 through 4-5 show address information formats. The information is valid for 6 CPs. The Address signals follow the leading edge of the Address Ready signal by 2 CPs.

- Address Parity signals - The Address Parity signals provide odd parity for each group of 4 address bits as follows:

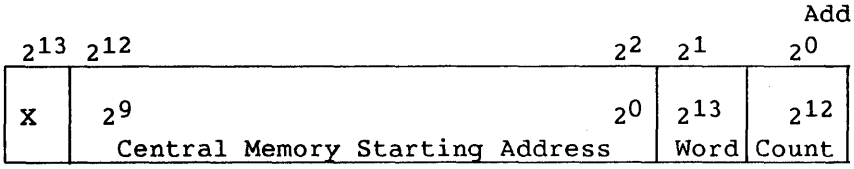
Parity Bit	Information Bits
2^0	2^0 through 2^3
2^1	2^4 through 2^7
2^2	2^8 through 2^{11}
2^3	2^{12} through 2^{15}

The Address Parity signals have the same timing as address signals and are valid for 6 CPs.

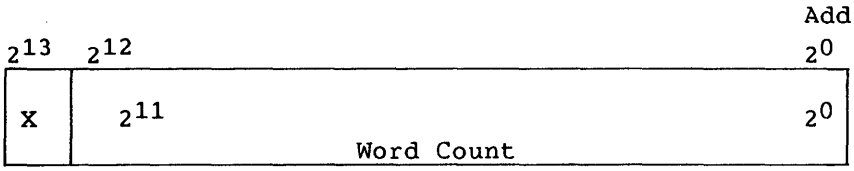
First address transfer:



Second address transfer:



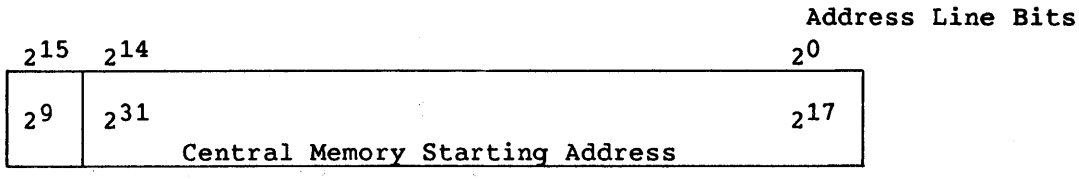
Third address transfer:



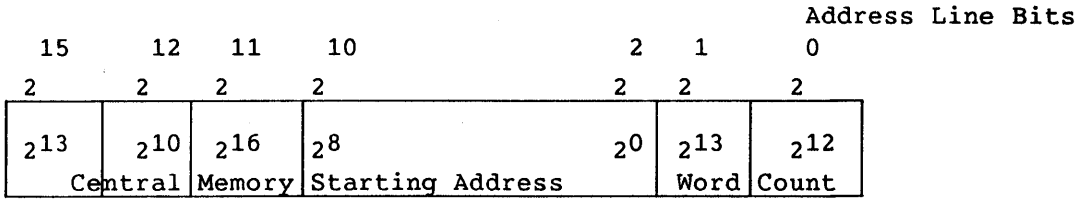
X indicates bits not used

Figure 4-3. Address and Word Count Format for CRAY X-MP Central Memory

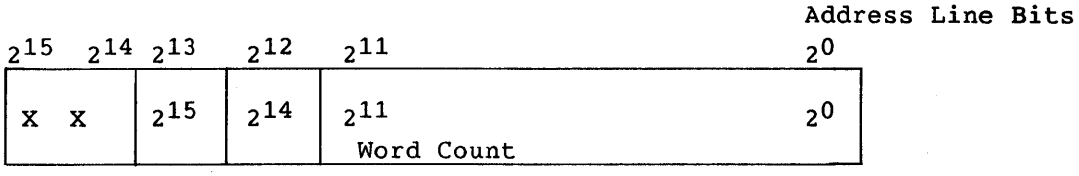
First address transfer:



Second address transfer:



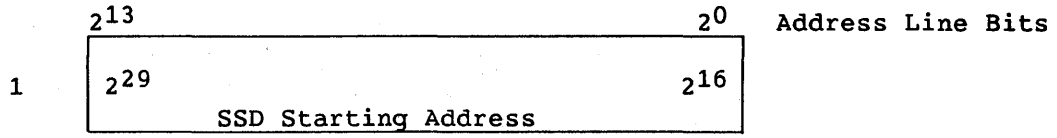
Third address transfer:



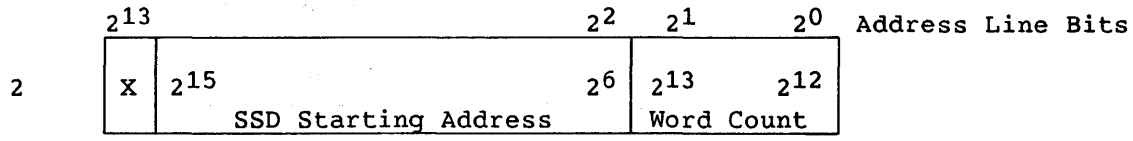
X indicates bits not used

Figure 4-4. Address and Word Count Format for CRAY X-MP EA Central Memory

First address transfer



Second address transfer



Third address transfer

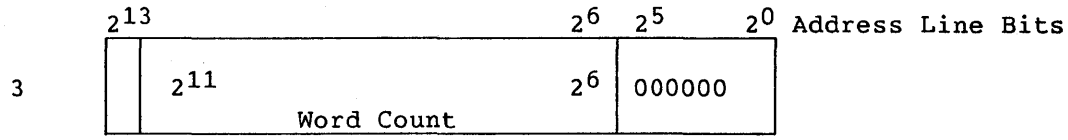


Figure 4-5. Address and Word Count Formats for an SSD

Central memory and SSD input channel interface registers

The following registers control transfers through the central memory input channel:

- A 16-bit local memory starting address register, loaded by an HIA : 1 function.
- The 24-bit central memory starting address register or a 30-bit SSD starting address register. Functions HIA : 2 and HIA : 3 each load portions of the address.
- A 14-bit block length register, loaded by an HIA : 4 function.

Central memory and SSD input channel functions

The following are functions for the central memory input channel:

Function	Description
HIA : 0	Clears the busy and done flags. This function causes an error condition if issued while the channel is active. An HIA : 0 function is the only method of clearing an error condition on the channel.

Function	Description
HIA : 1	Enters the contents of the IOP accumulator into the local memory starting address register. Hardware forces bits 2^0 and 2^1 of the address to 0. This function does not affect the busy or done flags. If this function is issued while the channel is active, an error condition occurs.
HIA : 2	Enters the starting address as follows: • When the channel is connected to central memory of an CRAY X-MP computer system, enters accumulator bits 2^0 through 2^{14} into the 2^9 through 2^{23} positions of the central memory starting address register. • When the channel is connected to central memory of an CRAY Y-MP or CRAY X-MP-EA computer system, enters accumulator bits 2^0 through 2^{15} into bits 2^{16} through 2^{31} of the central memory starting address register. • When the channel is connected to an SSD, enters accumulator bits 2^0 through 2^{14} into the 2^{15} through 2^{29} positions of the SSD starting address register. This function does not affect the busy or done flags. If this function is issued while the channel is active, an error condition occurs.
HIA : 3	Enters the starting address as follows: • When the channel is connected to central memory of an CRAY X-MP computer system, enters accumulator bits 2^0 through 2^8 into bits 2^0 through 2^8 of the central memory starting address register. • When the channel is connected to central memory of an CRAY Y-MP or CRAY X-MP-EA computer system, enters accumulator bits 2^0 through 2^{15} into bits 2^0 through 2^{15} of the central memory starting address register. • When the channel is connected to an SSD, enter accumulator bits 2^0 through 2^8 into bits 2^6 through 2^{14} of the SSD starting address register.

Function	Description
	This function does not affect the busy or done flags. If this function is issued while the channel is active, an error condition occurs. NOTE: When the channel is connected to an SSD, the 24-bit SSD address entered through the HIA : 2 and HIA : 3 functions is the SSD starting address divided by 100_8.
HIA : 4	Enters accumulator bits 2^0 through 2^{13} into the block length register and initiates a transfer of a block of data. The value in the block length register indicates the number of 64-bit words in a transfer. A zero value indicates the maximum of 16,384 words. The HIA : 4 function sets the busy flag, clears the done flag, and then initiates the transfer. When the transfer is complete, the done flag sets and the busy flag clears. If an unrecoverable error occurs during the transfer, the transfer terminates, the done flag sets, and the busy flag remains set. If this function is issued while the channel is active, an error condition occurs. Allow 1 CP before checking the busy or done flags. NOTE: When the channel is connected to an SSD, bits 2^0 through 2^5 of the word count must be 0's to avoid a block length error.
HIA : 6	Clears the interrupt enable flag for the channel. This function does not alter the busy or done flags.
HIA : 7	Sets the interrupt enable flag for the channel. This function does not alter the busy or done flags.
HIA : 14	Enters bits 2^0 through 2^2 of the IOP accumulator into the diagnostic mode register. These bits contain an octal digit that designates the diagnostic mode. Table 4-4 lists mode designators and their functions. Only one mode is valid at a time. All diagnostic modes are cleared with a master clear signal or by an HIA : 0 function followed by HIA : 14 function with the accumulator equal to 0. Mode 0 remains active until another HIA : 14 function selects another mode. This function is used for maintenance purposes only and is only available in maintenance mode.

Table 4-4. Memory Input Channel Diagnostic Modes

Mode Designator	Function
0	Sets the Last Word Flag signal (only if the channel is active).
1	Disables the Last Word Flag signal from external memory.
2	Forces a constant Address Ready signal.
3	Disables the third Address Ready signal.
4	Transfers the first address with parity bits forced to 0, the third address with parity bits forced to 1, and disable the Transmit Data signal to central memory.
5	Forces a Clear Channel signal with the IOP active.
6	Disables block length = 0.
7	Disables SECDED.

Central memory and SSD input channel error processing

The busy and done flags set if an unrecoverable error occurs in an input transfer from central memory. The channel interface generates an error code and sends it to an error log. Table 4-5 lists and describes the error codes. Single-bit errors do not affect the busy and done flags, because the SECDED circuitry corrects these errors automatically. However, the channel interface sends single-bit error syndrome information to the error log.

Table 4-5. Input Channel Error Codes

Error Code Designator	Error Code	Description
0	Single-bit error	A single-bit memory error was discovered and corrected.
1	Function error	A function 0, 1, 2, 3, or 4 was issued while the channel was active.
2	Active error	The central memory side of the interface became inactive while the IOP side was still active.
4	Central memory address error	The central memory side of the interface received more or less than three Address Ready signals, or a parity error occurred in one of the address channel transfers.
5	Central memory data error	The central memory side of the interface has a multiple data error from memory, or the IOP side received data with a multiple-bit error.
6	Block length error	The IOP received the last word and the block length count was not 0, or the last word was not received and the block length count was equal to 0.

CENTRAL MEMORY AND SSD OUTPUT CHANNEL

The central memory output channels (channel 15 on IOS-C IOPs; channels 15 and 53 on IOS-D IOPs) transfer data from local memory to central memory or an SSD. The channels carry 64 bits of data in parallel with 8 check bits; all transfers take place in 16-word bursts. A 16-bit channel carries address information.

NOTE: References to central memory in the remainder of the central memory output channel subsection also apply to the SSD, unless otherwise noted.

The central memory channel output signal sequence is more complex than that on an ordinary DMA channel. The central memory output channel passes more control signals and more data protection information. Figure 4-6 illustrates the signal exchange between the IOP and central memory or an SSD.

The following paragraphs describe central memory output channel signals, sequences, and functions. Section 5 of this manual describes central memory channel operation in bypass mode.

Central memory and SSD output channel signals

The following paragraphs describe the memory channel output signals (from local memory to central memory or an SSD) and the signal timing. Table 4-6 shows the sequence of signals for transferring data from local memory to central memory.

- Data signals - Data transfers in 64-bit words over 64 lines. Data on the data lines is valid 2 CPs after the leading edge of the Data Ready signal and stays valid for 6 CPs.

- Check-byte signals - The Check-byte signals consist of 8 error-check bits sent by central memory to the IOP to verify data accuracy. The Check-byte signals are valid 2 CPs after the leading edge of the signals and stay valid for 6 CPs.

- Data Ready signal - The Data Ready signal indicates data on the channel is valid. The leading edge of the Data Ready signal precedes the Data signals by 2 CPs and the Data Ready signal stays valid for 2 CPs. The minimum period for Data Ready signals is 6 CPs.

- Last Word Flag signal - The Last Word Flag signal indicates that the last data word of the transfer is on the data lines and the IOP word count is 0. The signal is valid 2 CPs after the leading edge of the signal and stays valid for 6 CPs.

- Clear Channel signal - The Clear Channel signal clears all error conditions in the channel. It remains set a minimum of 8 CPs. The Address Ready signal lags the trailing edge of the Clear Channel signal by at least 8 CPs.

- Address Ready signal - The Address Ready signal precedes the address information. The leading edge of the Address Ready signal leads the address information by 2 CPs and remains valid for 2 CPs. The minimum period for sequential Address Ready signals is 6 CPs.

Table 4-6. Central Memory Output Channel Sequence

Step	IOP Action	Mainframe or SSD Action
1.	Performs no action.	Raises the Transmit Address signal.
2.	Activates the channel: a. Raises the Address Ready signal and performs the first address transfer. b. Raises the Address Ready signal and performs the second address transfer. c. Raises the Address Ready signal and performs the third address transfer.	Performs no action.
3.	Performs no action.	Raises the Transmit Data signal. Drops the Transmit Address signal.
4.	Transfers the first block of data in 64-bit words. Raises the Data Ready signal before each 64-bit transfer.	Stores the first block in buffer.
5.	Performs no action.	Raises the Transmit Data signal.
6.	Transfers the second block of data in 64-bit words. Raises the Data Ready signal before each 64-bit transfer.	Stores the first block in central memory. Stores the second block in buffer.
7.	Performs no action.	Raises the Transmit Data signal.
8.	Transfers the third block of data in 16 64-bit words. Raises the Data Ready signal before each 64-bit transfer.	Stores the last block in central memory.
9.	Performs no action.	Sends the Transmit Address signal.
10.	Sets the done flag.	Performs no action.

Three Address Ready signals initiate a data block transfer. The IOP sends the first Address Ready signal only when the Transmit Data signal is set. The second and third Adress Ready signals follow automatically. The first Address Ready signal precedes bits 2^{10} through 2^{23} of the central memory starting address or bits 2^{16} through 2^{29} of the SSD starting address. The second Address Ready signal precedes bits 2^{0} through 2^{9} of the central memory starting address or bits 2^{6} through 2^{15} of the SSD starting address and bits 2^{12} through 2^{13} of the transfer word count. The third Address Ready signal precedes bits 2^{0} through 2^{11} of the transfer word count. Bits 2^{0} through 2^{5} of the SSD transfer word count must always be 0s or an SSD transfer length error occurs.

- Address signals - The 14 address lines carry the central memory starting address and the transfer word count. The address signals become active 2 CPs after the leading edge of the Address Ready signal and remain active for 6 CPs. Figures 4-4 through 4-6 show the formats for each of the three address information transfers.

- Address Parity signals - The Address Parity signals provide odd parity for each group of 4 address bits as follows:

Parity Bit	Address Bits
2^{0}	2^{0} through 2^{3}
2^{1}	2^{4} through 2^{7}
2^{2}	2^{8} through 2^{11}

  The Address Parity signals have the same timing as Address signals which are valid for 6 CPs.

- Disable Error-correction signal - The Disable Error-correction signal is used for diagnostic purposes only. When connected to central memory, this signal disables the SECDED circuitry used on the data channel. When connected to an SSD, this signal prevents double-bit channel errors from generating an unrecoverable error signal. The SSD memory still invokes SECDED and data errors are still reported to the SSD error logger.

- Transmit Data signal - The Transmit Data signal indicates that central memory can accept a block of data from the IOP. This signal remains set as long as central memory is able to receive the next 16-word burst without overflowing its input data buffers.

  The IOP looks at the Transmit Data signal at the beginning of each data block. If central memory sets the Transmit Data signal while using 16 banks, the IOP transmits a 16-word data block. If central memory is using 8 banks, the IOP transmits an 8-word data block. The Transmit Data signal from an SSD causes the IOP to transmit a 16-word data block. No data transfers until the Transmit Data signal sets.

NOTE: Transfers vary according to the device connected to the central memory input channel:

- All data transfers from CRAY X-MP computer systems are in 16-word blocks, with the exception of the last block, which can be an odd length.

- The number of 64-bit words sent in the first block of a CRAY-1 S central memory transfer equals the number of banks (16 or 8), minus the number specified by bits 2^0 and 2^1 (16 banks) or by bit 2^0 (8 banks) of the central memory starting address. This procedure adjusts the blocks to 16-bank or 8-bank boundaries so that the next block begins with bank 0.

- The number of 64-bit words sent in the first block of an SSD transfer is 16 minus the number specified by bits 2^6 and 2^7 of the SSD starting address.

- Unrecoverable Error signal - The Unrecoverable Error signal indicates that an unrecoverable error occurred in the transfer at the central memory side of the interface. The channel becomes inactive and remains so until the IOP sends a Clear Channel signal to central memory. The following conditions cause unrecoverable errors:

  - Central memory detected an address parity error.

  - Central memory received the Data Ready signal after receiving fewer than three Address Ready signals.

  - Central memory received more than three Address Ready signals.

  - Central memory detected an uncorrectable data error on the channel.

  - The transfer word count equals 0 and the Last Word Flag signal is cleared.

  - The transfer word count does not equal 0 and the Last Word Flag signal is set.

  The Unrecoverable Error signal remains set until the IOP sends a Clear Channel signal.

- Address Error signal - The Address Error signal indicates the unrecoverable error was an address error caused by one of the first three error conditions previously listed. The IOP samples the Address Error signal when the leading edge of the Unrecoverable Error signal appears.

- Transmit Address signal - The Transmit Address signal indicates the central memory side of the interface is inactive and ready to receive an address from the IOP. The central memory side of the interface clears this signal after receiving three Address Ready signals and does not set it again until the transfer is complete. If central memory discovers an error condition, the Transmit Address signal clears and does not set again until the IOP sends the Clear Channel signal.

Central memory and SSD output channel interface registers

The following registers control transfers over the central memory output channel:

- A 16-bit local memory address register loaded by an HOA : 1 function.
- A 24-bit central memory starting address register loaded by functions HOA : 2 and HOA : 3, which each load portions of the central memory starting address information.
- A 14-bit block length register, loaded by an HOA : 5 function.

Central memory and SSD output channel functions

The following are functions for the central memory and SSD output channel:

Function	Description
HOA : 0	Clears the busy and done flags and causes an idle condition in the channel. This function causes an error condition if issued while the channel is active. It is the only function that can clear an error condition in the channel.
HOA : 1	Enters the contents of the accumulator into the local memory address register. Hardware forces bits 2^0 through 2^1 of the address to 0's. This function does not alter the busy and done flags. An error condition occurs if the function is issued while the channel is active.
HOA : 2	Enters accumulator bits 2^0 through 2^{14} into bits 2^9 through 2^{23} of the central memory starting address register or into bits 2^{15} through 2^{29} of the SSD starting address register. This function does not alter the busy and done flags. An error condition occurs if the function is issued while the channel is active.

Function	Description
HOA : 3	Enters accumulator bits 2^0 through 2^8 into bits 2^0 through 2^8 of the central memory starting address register or bits 2^6 through 2^{14} of the SSD starting address register. This function does not alter the states of the busy and done flags. An error condition occurs if the function is issued while the channel is active. NOTE: When the channel is connected to an SSD, the 22-bit SSD address entered through the HOA : 2 and HOA : 3 functions is the SSD starting address divided by 100_8.
HOA : 5	Enters accumulator bits 2^0 through 2^{13} into the block length register and initiates transfer of a block of data. The value in the block length register indicates the number of 64-bit words in the transfer. A zero value in the block length register indicates the maximum 16,384 words in the transfer. The HOA : 5 function sets the busy flag, clears the done flag, and initiates the data transfer. Upon completion of the transfer, the done flag sets and the busy flag clears. If an unrecoverable error occurs during the transfer, the transfer terminates, the done flag sets, and the busy flag remains set. An error condition occurs if the function is issued when the channel is active. NOTE: When the channel is connected to an SSD, bits 2^0 through 2^5 of the word count must be 0's to avoid a block length error.
HOA : 6	Clears the interrupt enable flag. This function does not alter the busy and done flags.
HOA : 7	Sets the interrupt enable flag. This function does not alter the busy and done flags.
HOA : 14	Enters accumulator bits 2^0 through 2^2 into the diagnostic mode register. Table 4-7 summarizes the central memory output channel diagnostic modes. This function is only for maintenance purposes. Only one mode is valid at a time, except mode 6; mode 6 is not cleared by entering another mode. All diagnostic modes are cleared with a master clear procedure or by an HIA : 0 function followed by an HIA : 14 function with the accumulator equal to 0. Mode 0 remains active until another HIA : 14 function selects another mode.

Table 4-7. Central Memory Output Channel Diagnostic Modes

Mode Designator	Function
0	Sets the Last Word Flag signal (only if the channel is active).
1	Disables the Last Word Flag signal to external memory.
2	Forces a constant Address Ready signal.
3	Disables the third Address Ready signal.
4	Transfers the first address with parity bits forced to 0 and the third address with parity bits forced to 1.
5	Forces the Clear Channel signal without inactivating the IOP.
6	Uses bits 2^{56} through 2^{63} of the first word as the second word check byte, bits 2^{56} through 2^{63} of the third word as fourth word check byte, and so on.
7	Disables SECDED at central memory, or disable reporting of unrecoverable data errors by the SSD input channel. SSD memory errors are still reported to the SSD error logger.

Central Memory and SSD output channel error processing

The busy and done flags set if an unrecoverable error occurs in an output transfer. The channel interface generates an error code and sends it to an error log. Table 4-8 describes the output channel error codes.

Single-bit errors do not affect the busy and done flags, as the SECDED circuitry corrects these errors automatically. However, the channel interface sends single-bit error syndrome information to the error log.

Table 4-8. Output Channel Error Codes

Error Code	Name	Condition
1	Function error	A function 0, 1, 2, 3, or 5 was issued while the channel was active.
2	Active error	The central memory side of the interface became inactive while the IOP side was still active.
4	Central memory address error	The central memory side of the interface received greater or less than three Address Ready signals, a parity error occurred in one of the address channel transfers, or central memory received data while active.
5	Central memory data block length error	The central memory received data with a multiple-bit error, the last word was received and the block length count was not 0, or the last word was not received and the block length count was equal to 0.

CONSOLE KEYBOARD CHANNEL

An IOP provides four input channels (channels 40, 42, 44, and 46) that can be connected to operator console keyboards. The input channels are paired with output channels that can be connected to operator console displays. Each keyboard connects to a separate input channel and all operate independently. Data transfers serially from the keyboard to a channel interface register.

The channel has either a 7- or 8-bit interface register that assembles the data for a character associated with a key depression. Channel function TIA : 10 reads the data into the IOP's accumulator.

Channel hardware sets the busy flag at the beginning of the transmission. The busy flag clears and the done flag sets when the data assembles in the interface register.

Each pair of I/O channels has a programmable baud rate generator circuit. A TIA : 3 function sets the baud rate for both channels. The accumulator contents specify the baud rate as shown in Table 4-9.

Table 4-9. Baud Rate Settings

Accumulator Bits 2^0 - 2^2	Description
0	Disable input and output
1	300 Bd
2	Disable input and output
3	1,200 Bd
4	2,400 Bd
5	4,800 Bd
6	9,600 Bd (default after master clear)
7	19.2 KBd

Mnemonics TIA through TID represent functions on the four channels available on an IOP for console keyboards. The functions for the first channel are described below; the functions for the three remaining channels are identical. The following are the functions for the console keyboard channel:

Function	Description
TIA : 0	Clears the done flag.
TIA : 3	Enters the accumulator contents into the baud rate select logic.
TIA : 6	Clears the channel interrupt enable flag. This function does not alter the busy and done flags.
TIA : 7	Sets the channel interrupt enable flag. This function does not alter the busy and done flags.
TIA : 10	Reads the contents of the channel interface register into the low-order 7 or 8 bit positions in the IOP accumulator and clear the high-order bits. This function transfers the data to the accumulator, then clears the busy flag and sets the done flag.

CONSOLE DISPLAY CHANNEL

Four IOP output channels (channels 41, 43, 45, and 47) connect to operator console displays. The output channels are paired with input channels that can be connected to operator console keyboards. Each display connects to a separate channel and all can operate independently.

Data transfers serially from a channel interface register to the display device. Channel function TIA : 3 for the corresponding console keyboard channel selects the baud rate of the display channel. The channel has either a 7- or 8-bit interface register that receives data from the IOP accumulator and transmits the data serially to the display device.

Mnemonics TOA through TOD represent the four channels available on an IOP for display consoles. The functions for the first channel are described in the following paragraphs; the functions for the remaining channels are identical.

Function	Description
TOA : 0	Clears the busy and done flags.
TOA : 6	Clears the channel interrupt enable flag. This function does not alter the channel busy and done flags.
TOA : 7	Sets the channel interrupt enable flag. This function does not alter the channel busy and done flags.
TOA : 14	Enters the low-order 7 or 8 bits of the IOP accumulator contents into the channel interface register. Initially, this request sets the busy flag and clears the done flag. It transmits the data from the interface register to the display device and then clears the busy flag and sets the done flag.

MAINFRAME/MAINTENANCE INPUT CHANNEL

Each IOP has one channel dedicated to receiving data from a mainframe I/O channel. Data is transferred over this channel in block mode directly into IOP local memory.

This channel (channel 50) connects to the maintenance control unit (MCU) on all IOPs or to the CRI mainframe from the buffer I/O processor (BIOP), disk I/O processor (DIOP), and auxiliary I/O processor (XIOP). A master I/O processor (MIOP) typically connects to the CRI mainframe through channel pair 20/21 (refer to the "Front-end Interface Input Channel" subsection in this section).

The 16-bit local memory address register contains the starting address for the next local memory reference. Input operations load data into local memory in bursts of 4 parcels. Hardware forces the 2 low-order bits of the local memory starting address to 0 when the address enters the local memory address register. The register address increases by four at the end of a transfer. If an input transfer stops on a boundary other than four, the final memory reference stores undefined parcels. The final memory address is equal to the address of the last defined parcel, plus 1.

The following are functions for the mainframe/maintenance input channel:

Function	Description
LIA : 0	Clears the busy and done flags and abort any transfer in progress.
LIA : 1	Enters the accumulator contents into the local memory address register (forcing the 2 low-order bits to 0) and start an input transfer from the mainframe. Initially, the busy flag sets and the done flag clears. The LIA : 1 function receives and stores data in 4-parcel bursts, until the parcel count is 0 or until a disconnect signal is received from the CRI mainframe I/O channel. Upon completion of the LIA : 1 function, the done flag sets and the busy flag clears.
LIA : 2	Enters the accumulator contents into the channel interface parcel count register. The value in the parcel count register is a count of the parcels in a transfer. This function does not affect the state of the busy and done flags.
LIA : 3	Clears all four parity error flags. Four parity bits protect each 16-bit parcel sent on a CRI mainframe I/O channel. A parity error in a 4-bit group causes one parity error flag to set. A parity error does not affect the status of the busy and done flags. Issuing this function does not affect the busy or done flags.
LIA : 4	Clears the ready-waiting flag. The channel sets a ready-waiting flag when it is inactive and receives a Ready signal from the CRI mainframe. The LIA : 4 function discards the Ready signal by clearing the ready-waiting flag. The Ready signal must be discarded to resynchronize the transfer. This function does not affect the busy and done flags.
LIA : 6	Clears the interrupt enable flag. Monitor the channel through the busy and done flags when channel interrupts are disabled. This function does not affect the the busy and done flags.
LIA : 7	Sets the interrupt enable flag. The channel interrupts whenever the channel done flag sets. This function does not affect the busy and done flags.
LIA : 10	Transfers the contents of the local memory address register into the accumulator. The address is one greater than the address of the last parcel stored. This function does not affect the busy and done flags.

LIA : 11 Reads the contents of the status register into the accumulator. The status register contains the parity error flags and the ready waiting flag. Table 4-10 lists status register bit assignments.

Table 4-10. Mainframe/Maintenance Channel Status Register

Bit	Description
2^0	Always 0 after an LIA : 11 function issues
2^1	Always 0 after an LIA : 11 function issues
2^2	Parity error on channel 20, 22, 24, 26, 30, 32, 34, or 36
2^3	Always 0 after an LIA : 11 function issues
2^{15}	Ready-waiting flag

MAINFRAME/MAINTENANCE OUTPUT CHANNEL

Each IOP has one channel for sending data to a CRI mainframe 6 Mbyte/s I/O channel. Data is transferred in block mode directly from IOP local memory.

This channel (channel 51) interfaces to the MCU on all IOPs or it connects to the CRI mainframe from the BIOP, DIOP, or XIOP. On the MIOP, the CRI mainframe connection is typically through channel pair 20/21 (refer to the "Front-end Interface/Operator Workstation Input Channel" subsection in this section).

The 16-bit local memory address register contains the starting address for the next local memory reference. An LOA : 1 function request loads the register with the local memory starting address before initiating the output transfer. Hardware forces the 2 low-order bits of the starting address to 0 when the address is entered in the register because memory references are in bursts of 4 parcels. When the memory reference is complete, the register address is increased by 4. When the output transfer is complete, the local memory address in the register is one greater than the address of the last parcel sent.

The following are the functions for the mainframe/maintenance output channel:

Function Description

LOA : 0 Clears the busy and done flags and aborts any transfer. The busy and done flags are not valid until 2 CPs after the function is complete.

Function	Description
LOA : 1	Loads the current accumulator contents into the local memory address register (forcing the 2 low-order bits to 0) and start the transfer to a CRI mainframe. Initially, the busy flag sets and the done flag clears. The interface makes a memory reference starting at the address contained in the local memory address register and outputs 4 parcels of data. Then another reference reads out another burst of 4 parcels. When the parcel count stored in the register is reached, the transfer stops, the done flag sets, and the busy flag clears.
LOA : 2	Enters the accumulator contents into the interface parcel-count register. The parcel-count register holds a positive count of the number of parcels to be transferred. This function does not affect the busy and done flags.
LOA : 3	Clears the sequence-error flag. The sequence-error flag sets when the channel interface receives a Resume signal from the CRI mainframe and the interface is not busy or a local memory reference is in progress. If the interface interrupt enable flag is set, the sequence-error flag causes an interrupt. The sequence-error flag and the LOA : 3 function have no affect on the busy and done flags.
LOA : 4	Sends one of three control signals to a CRI mainframe. Accumulator bits 2^8, 2^9, and 2^{14} select the signal, as the following list shows:

Bit	Description
2^8	This bit sends a Disconnect signal to the CRI mainframe.
2^9	This bit stops the Automatic Disconnect signal that is sent at the end of a transfer.
2^{14}	This bit sends a Master-clear signal to the CRI mainframe.

An interface register holds the accumulator signal select bits until another LOA : 4 function enters new signal select bits. This function does not affect the busy and done flags.

Function	Description
LOA : 6	Clears the channel interrupt enable flag to prevent the interface from interrupting the IOP. This function monitors the channel through the busy and done flags or through the status information returned by an LOA : 11 function when channel interrupts are disabled. This function does not affect the busy and done flags.
LOA : 7	Sets the interrupt enable flag to allow interrupting the IOP. With interrupts enabled, the interface interrupts whenever the done flag sets or whenever a sequence error occurs. This function does not affect the busy and done flags.
LOA : 10	Enters the contents of the local memory address register into the IOP's accumulator. The address entered is one greater than the address of the last parcel transferred from local memory. This function does not affect the busy and done flags.
LOA : 11	Reads selected bits of the channel interface status register into the IOP accumulator. Bits 2^0 and 2^1 set to indicate the IOP number. Bit 2^{15} sets to indicate a sequence error. This function does not affect the busy and done flags.

NONSTANDARD IOP CHANNEL INTERFACES

This subsection describes the channel interfaces that are not assigned standard channel numbers. Nonstandard channel interfaces are available for the peripheral expander, high-speed graphic displays, front-end computers, block multiplexer channels, and disk storage units (DSUs). The Disk Systems Hardware Reference Manual, CRI publication number HR-0077, describes the operation of DSU channels. Section 6 of this manual lists suggested channel assignments for non-standard channel interfaces on the MIOP, BIOP, DIOP, and XIOP.

PERIPHERAL EXPANDER CHANNEL

An MIOP in a typical IOS-C configuration has a channel connected to a peripheral expander, which contains up to 16 controllers for peripheral devices including an 80-Mbyte disk drive, a magnetic tape unit, a printer/plotter, and an external clock.

Only one controller in the peripheral expander is active at a time, but the active controller services more than one peripheral device. Each peripheral unit has a unique device address. Channel functions select

individual peripheral units using the device addresses. Each controller has its own busy and done flags.

Interface registers

Each controller has three interface registers (A, B, and C). These registers together with the channel functions control data communication between the peripheral device and the IOP. The controller defines specific uses of the registers.

Channel assignments

All peripheral devices connected to one peripheral expander share the same channel number. Channel functions use device addresses to select individual peripheral units.

Delayed functions

Several functions for the peripheral expander channel require the peripheral expander data bus. These are delayed functions because they may have to wait for the data bus to be available. The channel interface indicates that a delayed function is complete by setting the done flag. The IOP does not send a delayed function before receiving the done flag for the previous delayed function.

The following are the delayed functions: EXB : 1, EXB : 2, EXB : 3, EXB : 4, EXB : 6, EXB : 14, EXB : 15, EXB : 16, and EXB : 17.

Transfer speeds

The peripheral expander interface has a maximum transfer rate of 16 Mbits/s from the IOP to the expander chassis. The interface has a maximum transfer rate of approximately 14.5 Mbits/s in the reverse direction in data channel mode or programmed I/O mode.

Block Size

The peripheral expander channel supports block transfers of $200,000_8$ parcels; however, the tape and printer controllers are limited to reading and writing $100,000_8$ parcels.

Peripheral expander channel functions

The following are the functions for the peripheral expander:

Function	Description
EXB : 0	Clears the peripheral expander busy, done, channel interrupt enable, and DMA enable flags. This function leaves the peripheral expander channel inactive and DMA references through the data channel disabled.
EXB : 1	Requests the contents of the A input register from the selected controller. Initially, the peripheral expander busy flag sets and done flag clears. This function uses the data bus and requires a function delay time of at least 1 microsecond. When the function completes, the done flag sets and the busy flag clears. An EXB : 1 function followed by an EXB : 10 function loads the A input register data into the IOP accumulator.
EXB : 2	Perform this function exactly as the EXB : 1 function, except request a transfer of the controller B input register contents.
EXB : 3	Perform this function exactly as the EXB : 1 function, except request a transfer of the controller C input register contents.
EXB : 4	Returns the status of the specified controller busy and done flags to the channel interface. This function determines which peripheral controller has the highest priority interrupt, which has its done flag set, and which has its interrupts enabled. This is a delayed function because it uses the bus data lines from the peripheral expander. The peripheral expander busy flag sets and the done flag clears 1 CP after this function issues. Upon completion of the function, the peripheral expander busy flag clears and the done flag sets. Issue an EXB : 11 function following the EXB : 4 function to load the controller busy flags, done flags, and the present interrupt device address.
EXB : 5	Loads accumulator bits 2^0 through 2^5 into the channel interface device address register. The device address register holds the device address of a controller to which a delayed function is being sent. This function loads the device address into the register in the clock period following instruction issue. The function does not affect the peripheral expander done and busy flags.

Function	Description
EXB : 6	Sends the accumulator contents to the peripheral expander. The 16-bit word from the accumulator acts as a mask to disable interrupts from specific controllers. Each bit in the mask word corresponds to a controller in the peripheral expander. If the mask bit is set, interrupts from the corresponding controller are disabled. Refer to the latest documentation for the peripheral controller to identify specific mask bits for specific devices. Initially, the peripheral expander busy flag sets and the done flag clears. This is a delayed function because it sends the mask throughout the peripheral expander by way of the bus data lines. Upon completion of the function, the peripheral expander busy flag clears and the done flag sets. This function clears the channel interrupt enable flag. If an interrupt condition is still present 400 ns after completion, the channel interrupt enable flag sets again.
EXB : 7	Enables or disables interrupts for the peripheral expander channel and the controllers according to the contents of accumulator bits 2^0 and 2^1. Bit 2^0 enables or disables peripheral expander interrupts (0 = disable; 1 = enable). Bit 2^1 enables controller interrupts (0 = disable; 1 = enable). This function does not affect the peripheral expander busy and done flags.
EXB : 10	Reads the data on the data bus into the IOP accumulator. Use this function to retrieve data received by the EXB : 1, 2, or 3 input functions. The data remains valid until another EXB : 1, 2, or 3 function is issued. This function does not affect the peripheral expander busy and done flags.
EXB : 11	Reads the peripheral expander channel status and the controller status into the IOP accumulator. The status word includes the interrupt device code number. Table 4-11 shows the status format for this function. This function does not affect the busy and done flags.

Table 4-11. EXB : 11 Status Format

Bit	Description
2^0	Interrupting device code bit 2^0
2^1	Interrupting device code bit 2^1
2^2	Interrupting device code bit 2^2
2^3	Interrupting device code bit 2^3
2^4	Interrupting device code bit 2^4
2^5	Interrupting device code bit 2^5
2^6	Unassigned
2^7	DMA enabled
2^8	Peripheral expander interrupts enabled
2^9	Controller interrupts enabled
2^{10}	Function active - delayed function executing
2^{11}	Peripheral expander busy flag
2^{12}	Peripheral expander done flag
2^{13}	Interrupt request from peripheral interface
2^{14}	Select busy flag of addressed device
2^{15}	Select done flag of addressed device

Function	Description
EXB : 13	Read the peripheral expander channel status and the controller status into the IOP accumulator. The status word includes the contents of the device address register. Table 4-12 shows the status format for this function. This function does not affect the busy and done flags.

Table 4-12. EXB : 12 Status Format

Bit	Description
2^0	Device address bit 2^0
2^1	Device address bit 2^1
2^2	Device address bit 2^2
2^3	Device address bit 2^3
2^4	Device address bit 2^4
2^5	Device address bit 2^5
2^6	Unassigned
2^7	DMA enabled
2^8	Peripheral expander interrupts enabled
2^9	Peripheral controller interrupts enabled

Table 4-12. EXB : 12 Status Format (continued)

Bit	Description
2^{10}	Function active - delayed function being performed
2^{11}	Peripheral expander busy flag
2^{12}	Peripheral expander done flag
2^{13}	Interrupt request from peripheral interface
2^{14}	Select busy flag of addressed device
2^{15}	Select Done flag of addressed device

Function	Description
EXB : 14	Sends the contents of the IOP accumulator to the selected controller. Initially, this function sets the peripheral expander busy flag and clears the done flag. The data transfers into the A register of the controller indicated by the device address in the device address register. Upon completion, this function clears the peripheral expander busy flag and sets the done flag. This is a delayed function and requires the peripheral expander data bus.
EXB : 15	Sends the IOP accumulator contents to the B register of the selected controller. This function is identical to the EXB : 14 function except that data goes into the B register of the addressed controller instead of the A register.
EXB : 16	Sends the IOP accumulator contents to the C register of the selected controller. This function is identical to the EXB : 14 function except that data goes into the C register of the addressed controller instead of the A register.
EXB : 17	Sends one of four control signals to the controller. Accumulator bits 2^0 through 2^3 select the control signal. Table 4-13 lists the control signals and corresponding accumulator bits. Individual controllers use the signals differently; however, the I/O Reset signal clears all controllers. The peripheral expander busy flag sets and the done flag clears at issue time. Upon completion of the function, peripheral expander done flag sets and the busy flag clears. This is a delayed function and requires the peripheral expander data bus.

Table 4-13. Accumulator Bit Control Signals

Bit	Signal
2^0	Start (S)
2^1	Clear (C)
2^2	Pulse (P)
2^3	I/O reset (IORST)

HIGH-SPEED EXTERNAL COMMUNICATIONS CHANNEL

The high-speed external communications (HSX) channel consists of an input and an output channel for transferring data from an XIOP to an external device. Most XIOP configurations place the HSX interface at channels 16 and 17. Figure 4-6 shows the signals exchanged on the HSX input and output channels.

HSX channel operation resembles central memory channel operation. Normal data transfer takes place in bursts of sixteen 64-bit words. When the total data block is greater than 16 words, the transfer sequence is one or more 16-word bursts followed by a final burst of 16 words or less.

The HSX channel differs from the central memory channel in the following ways.

- HSX input and output channel transfers do not require address information for the external device.
- HSX input and output channels are independent and must be initialized separately.
- Channel protocols for the HSX input and output channels are symmetrical. This symmetry allows input-to-output loop-back operation for maintenance purposes.

Transfer rates

The nominal transfer rate for the HSX input and output channels is 853 Mbits/s. This rate corresponds to one 64-bit word for every 75 ns. Transfer rates for equipment vary, depending on the available bandwidth between the channel interface and internal memory.

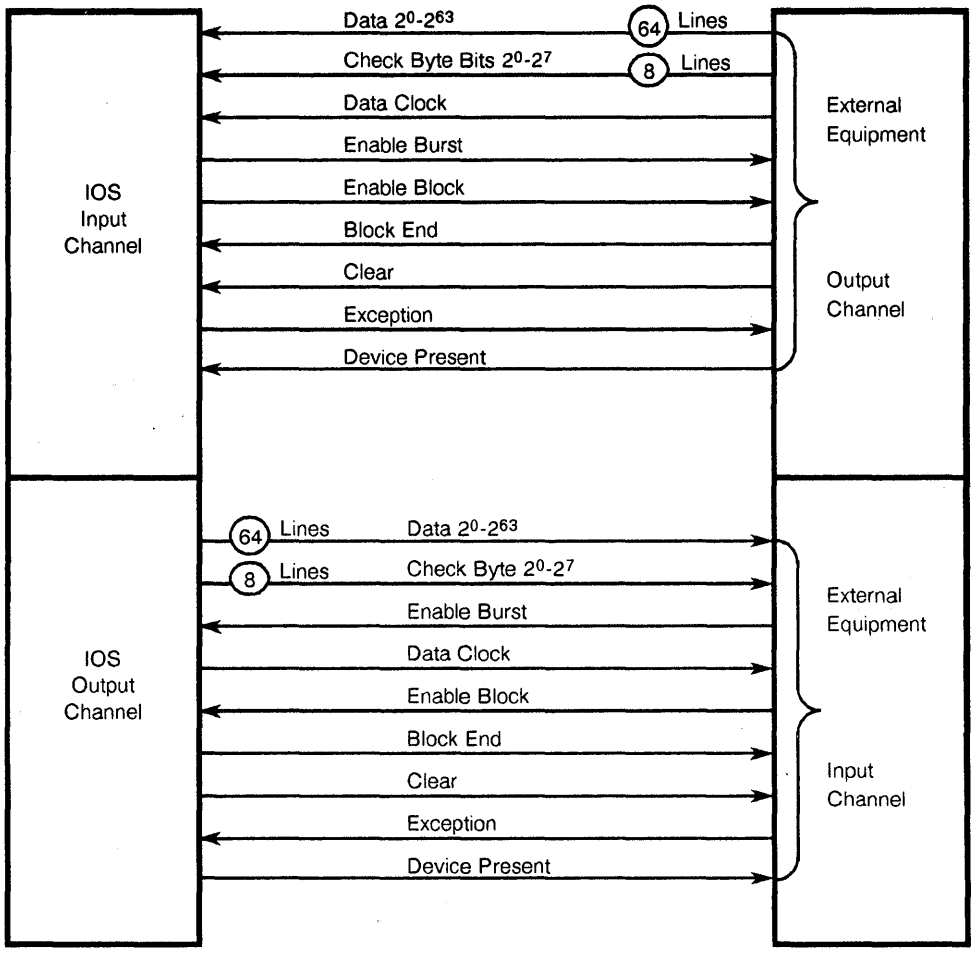


Figure 4-6. HSX Input and Output Channel Signals

Record size

HSX channel protocol does not limit record size. However, characteristics of the external device may impose restrictions on the record size of an individual transfer operation. To overcome these restrictions, chain two or more transfers together.

Input channel signals

Table 4-14 shows the sequence of the signals between an IOP and an external device on the input channel. The following paragraphs describe each signal.

- Data signals - Data transfers in 64-bit words over 64 lines. This signal is valid 2 CPs after the leading edge of the Data Clock signal and stays valid for 6 CPs.

- Check-bit signals - This 8-bit signal is encoded to provide SECDEC error protection for the data word. Timing for the Check-bit signals is identical to timing for the Data signals.

- Block End signal - This signal indicates the final data word of a block. The leading edge of the Block End signal occurs at the same time as the data transition for the final word. This signal remains set until the receiving device drops the Enable Block signal. With data chaining, the Block End signal remains cleared until the final word of data is sent.

- Data Clock signal - The external device activates this signal 25 ns before each new word of data is sent. Timing is measured at the external device cable end. The IOP samples the Data signal after the Data Clock signal transition.

  When the external device sends a new word of data every 75 ns, the Data Clock signal appears as a series of square waves with a 150-ns period. If the sending device transfers data words at varying intervals, the Data Clock signal will be irregular.

- Clear signal - This signal aborts channel transfer operations, resets channel hardware to an initialized state, and alerts the IOP when a condition that causes an Exception signal occurs in the external device. The IOP responds by dropping the Enable Block signal (if set) and issuing an Exception signal under program control. Data may be lost if a transfer is in progress when the Clear signal is sent.

- Device Present signal - The IOP uses this signal to determine if the external device is properly cabled, powered on, and enabled. The circuit for the Device Present signal is a steady-state current loop using two wires in the control part of the interconnect cables between the IOP and the external device. The IOP monitors the current flow to detect abnormal conditions such as a disconnected cable or sudden loss of power by the external device.

Table 4-14. HSX Input Channel Sequence

Step	IOP Action	External Device Action
1.	Activates the channel.	Performs no action.
2.	Raises the Enable Block signal.	Performs no action.
3.	Raises the Enable Burst signal.	Performs no action.
4.	Stores the first burst in I/O memory.	Transfers the first burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer.
5.	Raises the Enable Burst signal.	Performs no action.
6.	Stores the words in I/O memory.	Transfers the second burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer.
7.	Raises the Enable Burst signal.	Performs no action.
8.	Stores the final burst in I/O memory.	Transfers the final burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer. Raises the Block End signal.
9.	Drops the Enable Block signal.	Performs no action.

- Enable Block signal - The IOP asserts this signal to indicate that it is ready to receive a new block transfer. When the IOP receives the Block End signal, the Enable Block signal clears. The external device responds to the cleared Enable Block signal by clearing the Block End signal.

- Enable Burst signal - The IOP asserts this signal when it is ready for a new burst of up to 16 words of data. The external device checks the state of the Enable Burst signal before sending the first word of a burst. Once the burst begins, the sending device transmits the entire burst regardless of the state of the Enable Burst signal.

- Exception signal - This signal indicates that the IOP detects an error or acknowledges receipt of a Clear signal. It does not cause premature termination of a transfer operation or loss of data.

Input channel interface registers

The following registers control transfers over the HSX input channel:

- A local memory starting address register loaded by an XIA : 1 function

- A special functions register loaded by an XIA : 3 function

- A block length register loaded by an XIA : 4 function

Input channel functions

The following are functions for the HSX input channel:

Function	Description
XIA : 0	Clears the busy and done flags. This function causes an error condition if issued while the channel is active.
XIA : 1	Enters the IOP accumulator contents into the local memory starting address register and force bits 2^0 and 2^1 of the address to 0. This function does not alter the busy or done flags.
XIA : 3	Loads the following bits from the accumulator into the special functions register: • Bit 2^2 (send an Exeption signal) - When this bit equals 1, the IOP sends the external device an Exception signal. • Bit 2^3 (disable SECDED) - When this bit equals 1, the input channel ignores errors detected by SECDED. When this bit equals 0, SECDED operates normally.

Function	Description
XIA : 4	Enters accumulator bits 2^0 through 2^{13} into the block length register and initiate a transfer. This value indicates the number of 64-block words in a transfer. A zero value indicates the maximum of 16,384 words. The XIA : 4 function sets the busy flag, clears the done flag, and then initiates the transfer. When the transfer is complete, the done flag sets and the busy flag clears. If an unrecoverable error occurs during the transfer, the transfer terminates, the done flag sets, and the busy flag remains set.
XIA : 6	Clears the channel interrupt enable flag. This function does not affect the busy and done flags.
XIA : 7	Sets the channel interrupt enable flag. This function does not affect the busy and done flags.
XIA : 10	Reads the local memory address (LMA) register and send the contents to the accumulator.
XIA : 11	Reads the input status and load the following bits into the accumulator:

Bits	Description
2^0 - 2^7	All 0's
2^8	Device not present
2^{10}	Multiple bit error
2^{11}	Data overrun error
2^{12}	More data on channel
2^{13} - 2^{15}	All 0's

Input channel error processing

The channel interface sets the busy and done flags if a nonrecoverable (double-bit) error occurs. The interface generates an error code and sends it to the maintenance micro control unit (MMCU). Table 4-15 lists the input channel error codes. SECDED circuitry corrects single-bit errors automatically and sends syndrome information to an error log.

Table 4-15. Input Channel Error Codes

Error Codes	Description
0-4	These error codes are not used.
5	Multiple bit error. SECDED circuitry detected a double-bit error on the incoming data.
6	Data overflow error. A Write To Buffer signal is active and the current buffer (A or B) is full.
7	Both a multiple-bit error and data-overflow error occurred.

Output channel signals

Table 4-16 shows the sequence of the signals between an IOP and an external device on an output channel. The following paragraphs describe each signal.

- Data signals - Data transfers in 64-bit words over 64 lines. It is valid 2 CPs after the leading edge of the Data Clock signal and stays valid for 6 CPs.

- Check-bit signals - This 8-bit signal is encoded to provide SECDED for the data word. Timing for the Check-bit signals is identical to timing for the Data signals.

- Block-end signal - This signal indicates the final data word of a block. The leading edge of the Block-end signal occurs at the same time as the data transition for the final word. The signal remains set until the receiving device drops the Enable Block signal. With data chaining, the Block-end signal remains cleared until the final block of data transfers.

- Data Clock signal - The IOP activates this signal 25 ns before each new word of data is sent. Timing is measured at the external device cable end. The external device samples the Data signal after the Data Clock signal transition.

  When the IOP sends a new word of data every 75 ns, the Data Clock signal appears as a series of square waves with a 150-ns period. If the IOP transfers data words at varying intervals, the Data Clock signal is irregular.

Table 4-16. Central Memory Input Channel Sequence

Step	IOP Action	External Device Action
1.	Performs no action.	Activates the channel.
2.	Performs no action.	Raises the Enable Block signal.
3.	Performs no action.	Raises the Enable Burst signal.
4.	Transfers the first burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer.	Stores the first burst in I/O memory.
5.	Performs no action.	Raises the Enable Burst signal.
6.	Transfers the second burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer.	Stores the words in I/O memory.
7.	Performs no action.	Raises the Enable Burst signal.
8.	Transfers the final burst of data in 16 64-bit words. Raises the Data Clock signal before each 64-bit transfer. Raises the Block End signal.	Stores the final burst in I/O memory.
9.	Performs no action.	Drops the Enable Block signal.
10.	Drops the Block End signal.	Performs no action.

- Clear signal - This signal aborts channel transfer operations, resets channel hardware to an initialized state, and alerts the external device when an exceptional condition occurs in the external device. The external device responds by dropping the Enable Block signal (if set) and issuing an exception signal. Data may be lost if a transfer is in progress when the clear signal is sent.

- Device Present signal - The IOP uses this signal to determine if the external device is properly cabled, powered on, and enabled. The circuit for the Device Present signal is a steady-state current loop using two wires in the control part of the interconnect cables between the IOP and the external device. The IOP monitors the current flow to detect abnormal conditions such as a disconnected cable or sudden loss of power by the external device.

- Enable-block signal - The external device asserts this signal to indicate that it is ready to receive a new block transfer. When the IOP receives the Block-end signal, the Enable-block signal clears. The IOP responds to the cleared Enable-block signal by clearing the Block-end signal.

- Enable Burst signal - The external device asserts this signal when it is ready for a new burst of up to 16 words of data. The IOP checks the state of the Enable Burst signal before sending the first word of a burst. Once the burst begins, the IOP continues to transmit the remainder of the burst regardles of the state of the Enable Burst signal.

- Exception signal - This signal indicates that the the external device detected an error or acknowledges receipt of a clear signal.

NOTE: The Exception signal does not cause premature termination of a transfer operation or loss of data.

Output channel interface registers

The following registers control transfers over the HSX output channel:

- A local memory address (LMA) register loaded by an XOA : 1 function

- A special functions register loaded by an XOA : 3 function

- A block length register loaded by an XOA : 5 function

Output channel functions

The following are functions for the HSX output channel:

Function	Description
XOA : 0	Clears the busy and done flags and causes an idle condition in the channel. This function is required when clearing an error condition from the channel. Bit 20 of the special functions register must contain a 0 when the function issues to avoid terminating chained transfers.
XOA : 1	Enters the accumulator contents into the local memory address register and forces address bits 20 and 21 to 0. This function does not affect the busy and done flags.
XOA : 3	Loads the following bits from the accumulator into the special functions register: • Bit 2^0 (enable chaining) - When this bit equals 0, the channel terminates a block transfer operation by sending the Block End signal. When this bit equals 1, the channel suppresses the Block End signal, and the next block transfer becomes a continuation of the current transfer. • Bit 2^2 (send the Clear signal) - When this bit equals 1, the IOP sends a 75-ns Clear signal to the external device. When this bit equals 0, it does not select any special functions. The Clear signal aborts a data transfer and can be issued at any time. • Bit 2^3 (select an alternate check-byte mode) - When this bit equals 1, bits 2^{56} through 2^{63} of each even-numbered data word are sent as check bits for each odd-numbered data word. When this bit equals 0, check bytes for each word are generated normally.
XOA : 5	Enters accumulator bits 2^0 through 2^{13} into the block length register. This value indicates the number of 64-bit words in a transfer. A zero value indicates the maximum of 16,384 words. The interface sets the busy flag and clears the done flag when initiating the transfer. When the transfer is complete, the interface sets the done flag and clears the busy flag. If a nonrecoverable error occurs during the transfer, the transfer terminates, the done flag sets, and the busy flag remains set.

Function	Description
XOA : 6	Clears the channel interrupt enable flag. This function does not affect the busy and done flags.
XOA : 7	Sets the channel interrupt enable flag. This function does not affect the busy and done flags.
XOA : 10	Reads the LMA register. This function loads the contents of the LMA register into the accumulator.
XOA : 11	Reads the output status. This function loads the following status information into the accumulator:

Bits	Description
2^0	Exception pulse is recieved
2^1	Receiving device is aborted
2^2-2^{15}	Are all 0's

Output channel error processing

A nonrecoverable (double-bit) error in an output transfer causes the external device to send an Exception signal. The busy and done flags set at the end of the transfer. The interface generates an error code of 5 and sends the error code to the system error logger. SECDED circuitry corrects a single-bit error automatically.

FRONT-END INTERFACE/OPERATOR WORKSTATION INPUT CHANNEL

An MIOP has one or more channels dedicated to receiving data from a CRI mainframe, a CRI front-end interface, or a VME operator workstation.

These channels transfer data in block mode directly into IOP local memory. Typically, the IOS communicates with the CRI mainframe over MIOP channel pair 20/21. Channels 22 through 37 are available for front-end interfaces to other computer systems or Network Systems Corporation (NSC) adapters. Channel pair 16/17 or 26/27 on the IOS-D MIOP provide communication with a VME operator workstation.

The channel interface has a local memory address register that contains the starting address for the next local memory reference. A CIA : 1 function loads the register at the initiation of the input transfer.

Data transfers into local memory in bursts of 4 parcels. The local memory address register forces the 2 low-order bits of the address to 0, making the address a multiple of 4. When a reference is complete, the register address increases by 4. If the input transfer stops on another boundary, the final memory reference stores undefined parcels. The number of defined parcels equals the final memory address, which is the address of the last defined parcel, plus 1.

The following are functions for the front-end interface/operator workstation channel:

Function	Description
CIA : 0	Clears the busy and done flags and aborts any transfer in progress. Allow 1 CP before checking the busy or done flags.
CIA : 1	Enters the accumulator contents into the local memory address register (forcing the 2 low-order bits to 0) and starts an input transfer from the front-end. Initially, the busy flag sets and the done flag clears. The IOP receives and stores data in 4-parcel bursts until the parcel count is 0 or until the IOP receives a disconnect message from the channel interface. The transfer writes 0 data into memory if the transfer length is not a multiple of 4. When the transfer is complete, the done flag sets and the busy flag clears.
CIA : 2	Stores the accumulator contents in the channel interface parcel count register. The parcel count register holds a count of the parcels to be transferred. This function does not affect the busy and done flags.
CIA : 3	Clears all four parity error flags. Four parity bits protect each 16-bit parcel sent to the front-end I/O channel. A parity error in a 4-bit group sets one parity error flag. A parity error does not affect the busy and done flags. This function does not affect the busy or done flags.
CIA : 4	Clears the data waiting flag. The channel sets a data waiting flag if the channel is inactive and receives a data signal from the front-end. A channel operating in NSC mode accepts and buffers 4 data parcels. A channel operating in normal Cray channel mode accepts and buffers 1 parcel. In either mode, the buffered data goes into memory if the ready waiting flag is not cleared. A CIA : 4 function clears the data waiting flag and discards the data (4 parcels in NSC mode; 1 parcel in Cray channel mode). This function does not affect the busy and done flags.

Function	Description
CIA : 6	Clears the channel interrupt enable flag. When channel interrupts are disabled, monitor the channel using the busy and done flag status. This function does not affect the busy and done flags.
CIA : 7	Sets the channel interrupt enable flag. When channel interrupts are enabled, the channel interrupts the IOP whenever the done flag sets. This function does not affect the busy and done flags.
CIA : 10	Transfers the contents of the local memory address register into the accumulator. This value is equal to the address of the last parcel stored plus 1. This function does not affect the busy and done flags.
CIA : 11	Reads the contents of the status register into the accumulator. The status register contains three error flags. Table 4-17 lists the error flags and corresponding bit assignments.

Table 4-17. Front-end Interface Status Register

Bit	Error Flag
2^2	Channel parity error flag
2^{14}	Sequence error flag
2^{15}	Data waiting flag

FRONT-END INTERFACE/OPERATOR WORKSTATION OUTPUT CHANNEL

An MIOP configuration has one or more channels dedicated to sending data to a CRI mainframe or a CRI front-end interface. This channel transfers data in block mode directly from local memory. Typically, the IOS communicates with the CRI mainframe over MIOP channel pair 20/21. Channels 22 through 37 are available for front-end interfaces to other computer systems or NSC adapters. Channel pair 16/17 on the IOS-D MIOP provides communication with a VME operator workstation.

The channel interface has a local memory address register that contains the starting address for the next local memory reference. The COA : 1 function loads the register with the local memory starting address at the beginning of the output transfer. Data transfers from memory in bursts of 4 parcels. The local memory address register forces the 2 low-order bits of the address to 0, making the address a multiple of 4. When a reference is complete, the register address increases by 4. When output is complete, the local memory address in the register equals the address of the first parcel sent plus 1.

The following are functions for the front-end interface output channel:

Function	Description
COA : 0	Clears the busy and done flags and abort any transfer.
COA : 1	Enters the accumulator contents into the local memory address register (forcing the 2 low-order bits to 0) and start the transfer to the front-end. The IOP sends data in 4-parcel bursts, beginning at the address specified by the starting address register. The transfer continues until the parcel count is 0. When the transfer is complete, the done flag sets and the busy flag clears.
COA : 2	Enters the accumulator contents into the parcel count register. This value is a positive count of the number of parcels to be transferred. This function does not affect the busy and done flags.
COA : 3	Clears the sequence error flag. The sequence error flag sets if the channel interface receives a Resume signal from the front-end processor when the channel interface is not busy or a local memory reference is in progress. If the channel interrupt enable flag is set, the sequence error flag causes an interrupt. The sequence error flag and the COA : 3 function have no affect on the busy and done flags.
COA : 4	Sends control signals to a front-end interface. The following accumulator bits select the control signals:

Bit	Signal
2^8	Disconnect signal
2^9	Hold Disconnect signal (used during chaining)
2^{14}	Master Clear signal

The channel interface holds the accumulator signal select bits; another COA : 4 function alters them. This function does not affect the busy and done flags.

Function	Description
COA : 6	Clears the channel interrupt enable flag. Monitor the channel through the busy and done flags or through the status information returned by a COA : 11 function when channel interrupts are disabled. This function does not affect the busy and done flags.
COA : 7	Sets the channel interrupt enable flag. With interrupts enabled, the channel interface interrupts whenever the done flag sets or whenever a sequence error occurs. This function does not affect the busy and done flags.
COA : 10	Enters the contents of the local memory address register into the accumulator. The value in the local memory address register is 1 greater than the address of the last parcel transferred from local memory. This function does not affect the busy and done flags.
COA : 11	Reads the interface status into the accumulator. When set, bit 2^{15} of the status word indicates a sequence error. This function does not affect the busy and done flags.

BLOCK MULTIPLEXER CHANNELS

The IOS communicates with IBM-compatible equipment over block multiplexer channels from the XIOP.

Block multiplexer channels are grouped into 4-channel sets that share one Cray BMC-4 or BMC-5 block multiplexer controller (channel interface). Up to three controllers may be configured on an XIOP. Each controller interfaces the XIOP through a single DMA port. The speed capacity of each DMA port is shared by the four controller channels, which operate asynchronously and compete only for local memory references. A single channel drives one or more IBM-compatible control units. The IBM-compatible control units drive peripheral devices.

A BMC-4 channel operates in selector channel mode, byte multiplexer mode, or block multiplexer mode. A BMC-5 channel operates in selector channel mode, block multiplexer mode, or data-streaming mode. All IBM commands are possible for both BMC-4 and BMC-5 controllers. Both BMC-4 and BMC-5 channels have the following features:

- Command and data chaining
- Detection of retry status
- I/O error alert capability
- The high-speed data rate option

Refer to the appropriate IBM publication for detailed information and definitions of IBM terminology.

Transfer rates

Table 4-18 shows the approximate maximum data transfer rate and corresponding operational mode. In all modes except data-streaming mode, exact data rates are determined by cable length and signal turn-around time within the IBM control unit. In these modes, the BMC-4 or BMC-5 must receive an appropriate response signal or signals for each byte of data or control information sent. The high-speed option (requiring an additional pair of control lines) doubles the rate of data transfer in selector and block multiplexer modes.

Table 4-18. IBM Channel Transfer Rates

Protocol	Maximum Transfer Rate
Selector	0.80 Mbytes/s
Block multiplexer	0.80 Mbytes/s
Block multiplexer or selector high-speed option	1.6 Mbytes/s
Data streaming (BMC-5 only)	2.0, 3.0, or 4.0 Mbytes/s

Data transfer

The block multiplexer controller interfaces the 8-bit IBM channel to the 16-bit IOP local memory. The controller assembles data in the 8-bit IBM byte format into 16-bit parcels during input operations and disassembles the parcels into 8-bit bytes during output operations. Figure 4-7 shows block multiplexer controller data assembly and disassembly.

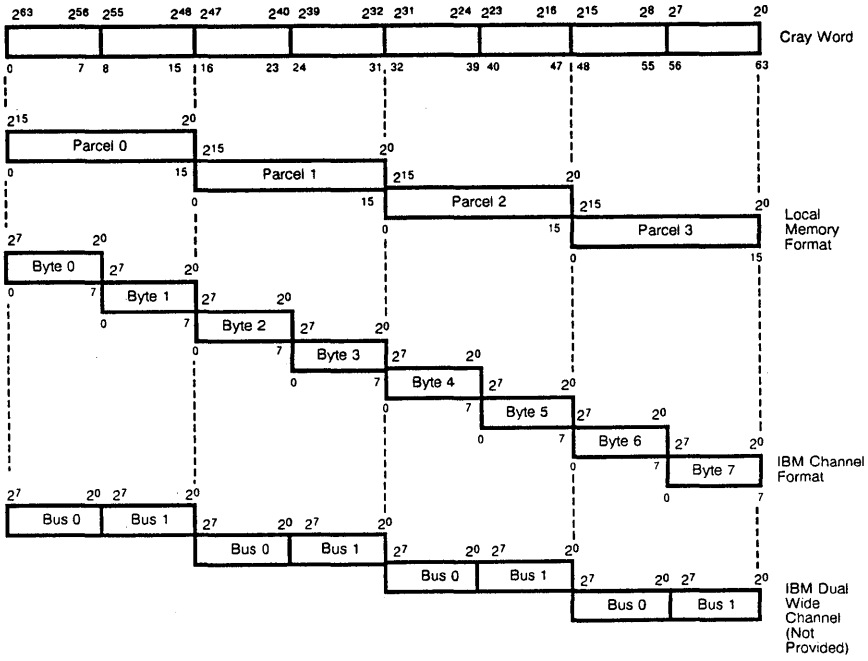


Figure 4-7. Block Multiplexer Controller Data Assembly/Disassembly

The controller reads four 16-bit parcels from local memory during an input operation and sends out eight 8-bit bytes to the block multiplexer channel. Four more 16-bit parcels are read into buffers on the controller while the channel transmits the first group of four 16-bit parcels transfered.

The controller reads eight 8-bit bytes from the channel and writes four 16-bit parcels into local memory during an output operation. If a read operation from the channel terminates with a byte length that does not assemble into 4 parcels, the final local memory write operation includes unpredictable data in the final parcels (enough to fill 4 parcels).

Record size

Data records can be any size except zero bytes. Use data chaining for record lengths greater than 65,535 bytes. The data chaining feature allows a single, large record to be broken into any size convenient for movement through local memory and for storage in buffer memory. The program controls this process using the BMC-4 or BMC-5 channel functions.

When the channel reads an IBM tape record, a short header field can be stored at one memory area, and the following data field can be stored at a different memory area, beginning at a different 64-bit word boundary.

Parity

An IOP checks all address, status, and data input lines for odd parity. An IOP generates odd parity for all address, status, and data-output lines.

Interrupts

Channel functions BMA : 6 and BMA : 7 enable or disable interrupts for individual block multiplexer channels. The channel sends an interrupt request when interrupts are enabled and when one of the following occurs:

- The done flag sets (on completion of functions BMA : 1 through BMA : 5).
- The request-in tag line becomes active, and an earlier BMA : 16 function selected request-in interrupt select mode.
- The byte counter decrements to 0 during data chaining and the last local memory reference is complete for that segment of data.

A BMA : 0 function clears the channel done and data chaining interrupts. It also clears the request-in tag interrupt if interrupts are first disabled by a BMA : 6 function. A BMA : 0 or BMA : 14 function clears data chaining interrupts after completion of a data transfer.

NOTE: Data chaining interrupts in the BMC-5 are not cleared by a BMA : 6 function.

Block multiplexer channel functions

The following paragraphs describe block multiplexer channel functions. Channel function descriptions apply to both the BMC-4 and the BMC-5 unless otherwise noted.

Function	Description
BMA : 0	Clears the busy and done flags and all output tags except operational out (OP-OUT) and suppress out (SUP-OUT). The function also clears interrupt conditions, if interrupts are disabled by a previous BMA : 6 function. Because the BMA : 0 function cannot be interlocked using the done flag, allow a 12-CP delay before issuing the next function to the channel.

Function	Description
BMA : 1	Performs one of several reset functions. Accumulator bits 2^1 and 2^0 select the specific reset function as shown in Table 4-19.

Table 4-19. Send Reset Function Parameters

Parameter 2^1 - 2^0	Function
0 0	Clears all output tag lines
0 1	Interface disconnect
1 0	Selective reset
1 1	System reset

Parameter xxxxx0 - Clears all output tag lines and bus 0 out lines. The channel initially clears the done flag and sets the busy flag. Upon completion, the channel sets the done flag and clears the busy flag.

Parameter xxxxx1 - Performs an interface disconnect function on the currently selected control unit.

The parameter initially clears the done flag and sets the busy flag. The currently selected control unit clears all input tag lines to the BMC-4 or BMC-5. The control unit continues with whatever sequence it was performing before the xxxxx1 command issued. When the control unit reaches a normal ending point in the sequence, it transfers the status it accumulated since the previous time it tranferred status.

The device path for the peripheral device remains busy from the time the interface-disconnect operation is initiated until the device-end status is accepted by the BMC-4 or BMC-5. The operation is completed in about 7 ms. Upon successful completion, the done flag sets and the busy flag clears.

If the Operational-in signal from the equipment does not clear within 6 ms of function issue, the IOP done flag sets and the busy flag remains set, indicating an error condition.

Function	Description
	Parameter xxxxx2 - Performs a selective reset function. Initially, the done flag clears and the busy flag sets. The control unit drops the operational-in tag and resets the currently selected peripheral device. The peripheral device status is also reset. The current operation continues until its normal stopping point and then stops. No data transfers after the stop. The selective reset operation does not affect the control unit ready status and only affects the peripheral device currently operating. The device end status is retained for transfer to the IOP channel after the reset. The selective reset operation is complete in about 7 ms. At this time, the busy flag clears and the done flag sets. Parameter xxxxx3 - Performs a system reset operation. The system reset operation clears the Operational-out signal, clears the status for each peripheral device, and resets all control units as well as the attached peripheral devices. The done flag clears and the busy flag sets at the beginning of operation. The operation is complete in about 6 ms. At this time the done flag sets and the busy flag clears. If one or more input tag lines are not reset at the end of the system reset operation, both the busy and done flags set, indicating an error condition.
BMA : 2	Sends a command to the control unit. Initially, the done flag clears and the busy flag sets. When the function completes, the busy flag clears and the done flag sets. Parameter command bits - Accumulator bits 2^0 through 2^7 select a command for an IBM-type control unit. Table 4-20 lists IBM commands and gives the corresponding bit assignments. The specific modifier codes and mode depend on the IBM-type control unit and the peripheral device. All command operations begin with an initial selection sequence that sends the command byte to the control unit. The initial selection sequence ends with an ending status (channel end or channel end with device end), a 0 status, or an error status. When the command calls for a transfer of data bytes, a zero status indicates that the transfer can begin.

Table 4-20. IBM Channel Command Bit Assignments

Command	P	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
Test	P	M	M	M	M	0	0	0	0
Test I/O	1	0	0	0	0	0	0	0	0
Sense	P	M	M	M	M	0	1	0	0
Sense ID	1	1	1	1	0	0	1	0	0
Read backward	P	M	M	M	M	1	1	0	0
Write	P	M	M	M	M	M	M	0	1
Read	P	M	M	M	M	M	M	1	0
Control	P	M	M	M	M	M	M	1	1

M = modifier bit
P = parity bit

Function Description

A transfer of data ends when the byte count decrements to 0 and no data chaining condition exists. The BMC-4 or BMC-5 indicates the function end by raising the service-out or data-out tag. The controller responds by raising the status-in tag.

The done flag sets when the control sequence processing is complete and the data (if required) is transferred to or from local memory. Figure 4-8 shows a typical channel sequence for a read operation to the IOP.

BMA : 3 (BMC-4)

Recognizes the request-in tag and reads the address of the device that made the request. Initially, this function clears the done flag and sets the busy flag. If the channel interface detects the request-in tag, it accepts the requesting address. The function checks the address for valid parity, sets the done flag, and clears the busy flag. If the interface does not detect the request-in tag, the function sets the done flag, and the busy flag remains set. This function ignores the address miscompare status (returned to accumulator bit 2^{12} after a BMA : 13 function) if the requesting address value is not known before the function issues. Figure 4-9 illustrates a typical request-in channel sequence.

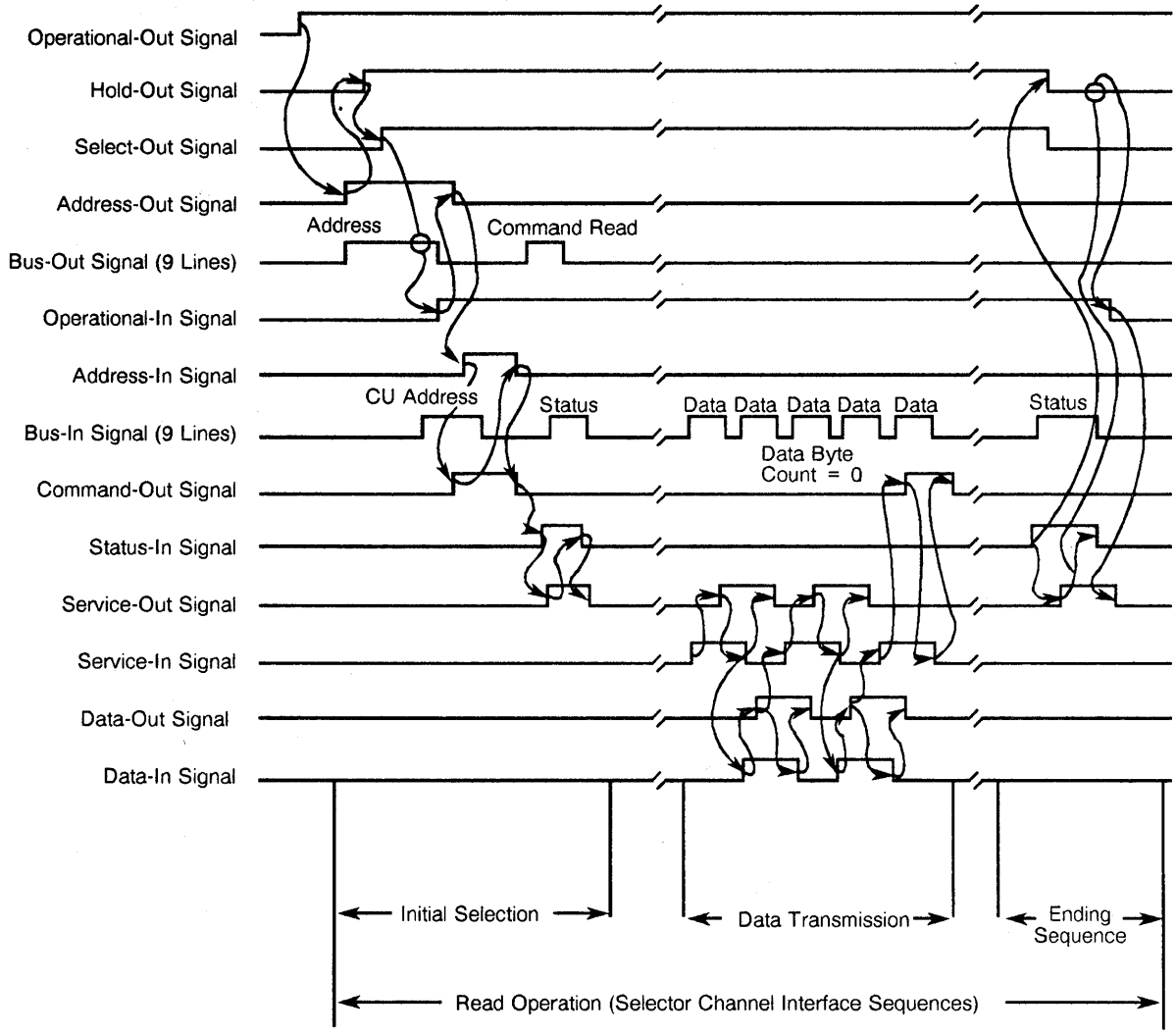


Figure 4-8. BMC-4 and BMC-5 Channel Read Sequence

NOTE: Clear and disable the request-in tag before issuing the BMA : 3 function if performing asynchronous data and status processing when interrupts are enabled. Clear and disable the request-in tag using the following sequence:

1. BMA : 6 - Clear the interrupt enable flag.
2. A = 0 - Clear the accumulator.
3. BMA : 16 - Disable request-in tag interrupts.
4. BMA : 0 - Clear interrupt conditions.

A constant interrupt occurs if this procedure is not followed.

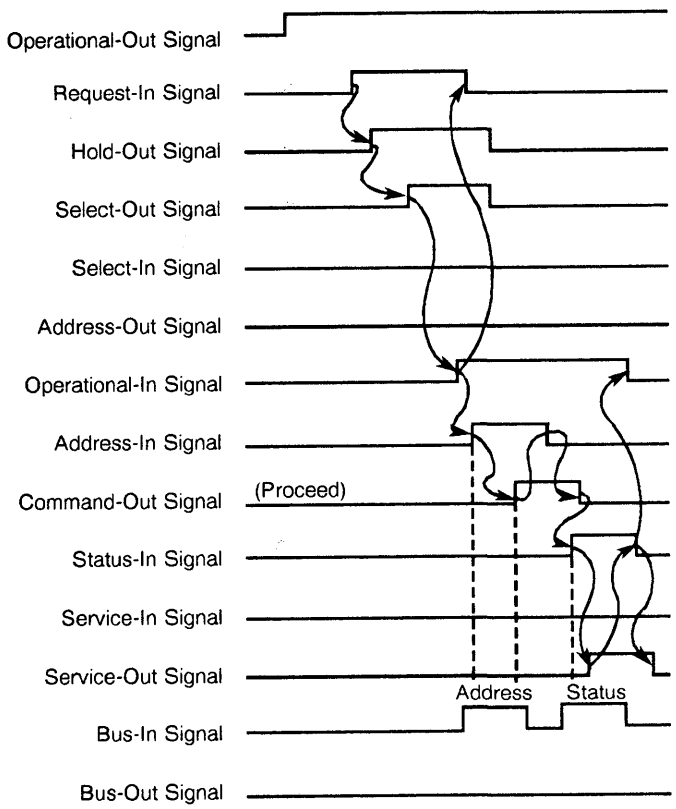


Figure 4-9. BMC-4 and BMC-5 Request-in Channel Sequence

Function	Description
	This function recognizes the request-in tag and reads the address of the device that made the request. Initially, this function clears the done flag and sets the busy flag. The function proceeds one of the following two ways depending on the contents of the accumulator: • If the accumulator equals 0, the channel checks the request-in tag. If the tag line is active, the sequence proceeds. If the tag line is not active, the function ends, the done flag sets, and the busy flag remains set. • If the accumulator equals 3, the channel waits for the request-in tag to become active. When the request-in tag becomes active, the sequence proceeds. The IOP software provides time-out protection to prevent indefinite waiting. The control unit presents the device address byte on the bus-0 in lines after the request-in line becomes active. The BMC-5 checks byte parity and stores the device address in the status register. The function terminates one of two ways, depending on the value of mode register bit 2^5 as shown below: • If mode bit 2^5 equals 0 and no parity error was detected, the operation ends with the done flag set and the busy flag cleared. • If mode bit 2^5 equals 1, the BMA : 4 function begins automatically. This function starts the transfer of the status and address to the IOP. Figure 4-9 illustrates a typical request-in channel sequence.
BMA : 4	Loads the accumulator with 1 byte of status information. Initially, the done flag clears and the busy flag sets. This function saves status in the status register if the control unit raises the status-in tag. If a previous BMA : 16 command set the stack status flag, the interface returns the command-out tag in place of the service-out tag to indicate that the channel should stop. When the sequence is complete, the done flag sets and the busy flag clears. Refer to Figure 4-10 for the asynchronous data and status processing sequence.

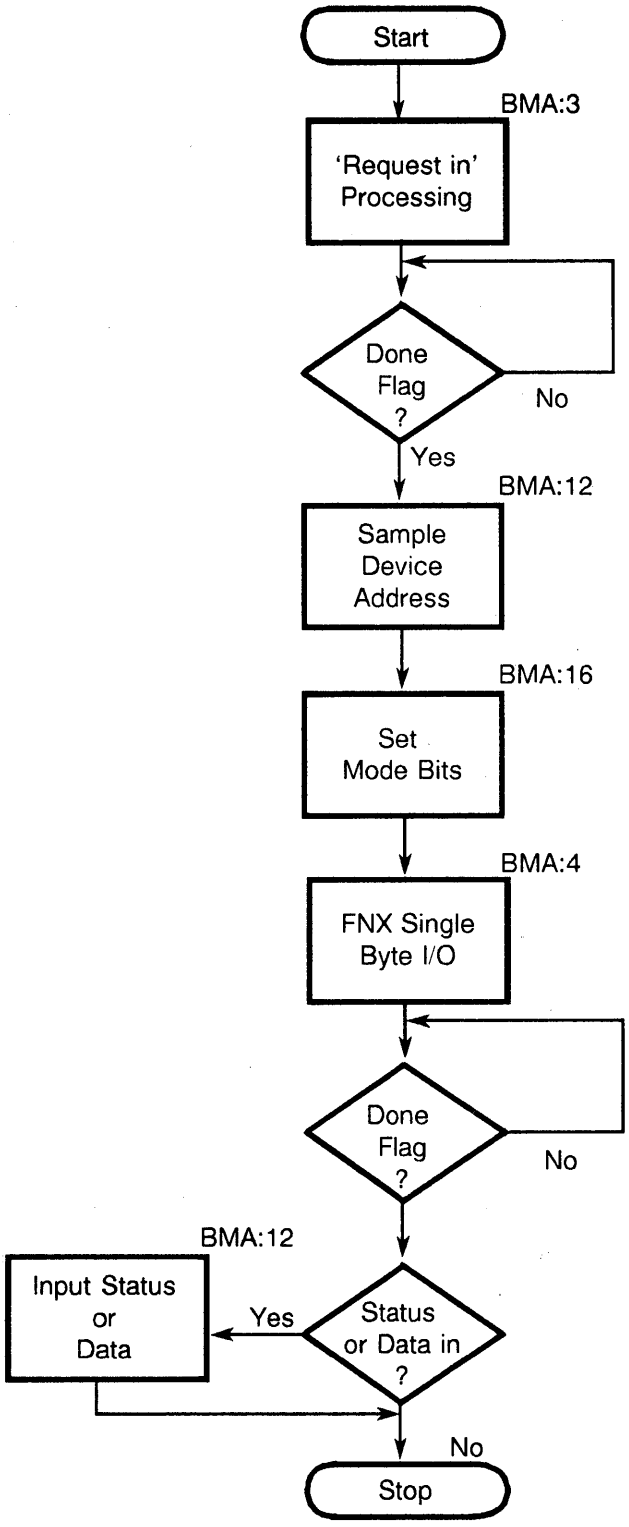


Figure 4-10. Asynchronous Data and Status Processing Sequence

Function	Description
BMA : 5 (BMC-4)	Performs a 10-microsecond delay procedure. Initially, the busy flag sets and the done flag clears. After the delay times out, the done flag sets and the busy flag clears.
BMA : 5 (BMC-5)	Performs one of four tests, depending on the contents of accumulator bits 2^0 and 2^1. This function is for maintenance purposes only. Initially, the done flag clears and the busy flag sets for all tests. The tests are listed in Table 4-21 and described in the following pages.

Table 4-21. BMC-5 Maintenance Diagnostic Tests

Accumulator Value	Test
*xxxxx*0	10-ms delay
*xxxxx*1	Tag line test
*xxxxx*2	Status conditions test
*xxxxx*3	Modes conditions test

Parameter *xxxxx*0 - 10-Microsecond delay

The purpose of the 10-microsecond delay test is to test the delay counter. After the delay expires, the function sets the done flag and clears the busy flag. An interrupt request is sent at the time of completion if interrupts are enabled.

Parameter *xxxxx*1 - Tag Line Test

This test raises the following output tag signals in response to corresponding input tag signals:

Input tag detected	Output tag set
Operational-in	Operational-out
Address-in	Address-out
Disconnect-in	Hold-out
Select-in	Select-out
Request-in	Command-out
Service-in	Service-out
Data-in	Data-out
Status-in	Suppress-out

The done flag sets to end the sequence.

Parameter xxxxx2 - Status Conditions Test

This test raises the following output lines in response to the corresponding status conditions:

Condition detected	Output set
Byte counter > 0	Operational-out
Command retry status	Address-out
Nonzero status byte	Hold-out
Error status	Select-out
Bus 0 parity incorrect	Command-out
Mode bit 2^5 present	All output tags
Mode bit 2^5 not present	Channel Busy

The done flag sets to end the sequence.

Parameter xxxxx3 - Mode Conditions Test

This test checks a series of mode-selection capabilities and responds with a series of output signals. The following are the tested modes and corresponding output signals.

Mode detected	Output set
Skip flag	Operational-out
Stack flag	Address-out
Command chain mode 1	Hold-out
Command chain mode 2	Select-out
Command chain mode 3	Command-out
Channel mode 1	Service-out
Channel mode 2	Data-out
Channel mode 3	Suppress-out

Function	Description
BMA : 6	Clears the channel interrupt enable flag. Monitor the channel through the done flag when channel interrupts are disabled. This function does not affect the busy and done flags.
BMA : 7	Sets the channel interrupt enable flag.
BMA : 10	Transfers the current value in one of two local-memory address registers into the accumulator. The state of accumulator bit 2^0 at function issue determines the register. The channel logic includes two local memory address registers to support data chaining. Table 4-22 lists the response bits returned to the accumulator.

Table 4-22. Read Local Memory Address Response Bits

Accumulator Bit	Description
2^0	Register select status: 0 = local memory address register 0 1 = local memory address register 1
2^1	Data-chaining flag status (1 = data chaining)
2^2	Local memory address 2^2
2^3	Local memory address 2^3
2^4	Local memory address 2^4
2^5	Local memory address 2^5
2^6	Local memory address 2^6
2^7	Local memory address 2^7
2^8	Local memory address 2^8
2^9	Local memory address 2^9
2^{10}	Local memory address 2^{10}
2^{11}	Local memory address 2^{11}

Table 4-22. Read Local Memory Address Response Bits (continued)

Accumulator Bit	Description
2^{12}	Local memory address 2^{12}
2^{13}	Local memory address 2^{13}
2^{14}	Local memory address 2^{14}
2^{15}	Local memory address 2^{15}

Function	Description
	Interrupts occur after each buffer transfer during data chaining. NOTE: Insert a single instruction delay (012000_8) after the BMA : 10 function if a BMA : 10, 14, 15, 16, or 17 function follows immediately. The delay ensures that the local memory address returned by the BMA : 10 function is for the expected channel. A programming restriction applies if the IOP instruction preceding a BMA : 10 function is any of the following: 4 through 7, 12, 13, 16, 17, 22, 23, 32, 33, 44 through 47, 52, 53, 62, or 63. In these cases, insert a logical product instruction 011 or 015 with the d or k field set to all 1s between the preceding instruction and the BMA : 10 instruction. This restriction is due to timing requirements in the adder and shifter.
BMA : 11	Reads the byte counter. A BMA : 15 function loads the byte counter initially. The byte counter decrements by 1 for each byte transferred. When the byte counter contains 0, the channel interrupts the IOP to terminate the transfer. If a control unit terminates the transfer, a nonzero value remains in the byte counter. During data chaining, the byte counter decrements to 0 and the channel sends an interrupt each time a segment transfers. A BMA : 15 function reloads the byte counter for the next segment transfer.

Function	Description
	Issue the BMA : 11 function twice in succession to get a current byte count to the accumulator. This requirement is due to timing restrictions in the IOP. The first execution moves the byte count to the byte counter status register. The second execution moves it to the accumulator. A BMA : 11 function followed by a BMA : 15 function operates as a diagnostic procedure to verify correct operation of the accumulator fanout, the intermediate byte counter status register, and the status path back to the accumulator. The BMA : 15 function loads the byte count into the byte counter status register. A subsequent BMA : 11 function reads the byte count from the byte counter status register to the accumulator. This function does not affect the busy and done flags. NOTE: Insert a single instruction delay (012000_8) after the BMA : 11 function if a BMA : 10, 14, 15, 16, or 17 function follows immediately. The delay ensures that the local memory address returned is for the expected channel.
BMA : 12	Reads status and address information. The status register in the channel interface holds the address and status mode bits read from the block multiplexer channel. Issue the BMA : 12 function twice in succession to get status and address information to the IOP accumulator. Two BMA 12 functions are required because of fixed timing in the IOP. The first execution moves the current status and address to the block multiplexer controller status register. The second execution moves the status and address to the IOP accumulator. A BMA : 12 function following a BMA : 16 function operates as a diagnostic procedure to verify correct operation of the accumulator fanout, the intermediate status register, and the status path back to the IOP accumulator. Issuing a BMA : 16 function loads address and mode bits into the status register. The BMA : 12 function reads the status back to the IOP accumulator. This function does not affect the busy and done flags. Table 4-23 lists status and address bits and corresponding accumulator locations.

Table 4-23. Status Register Bits

Accumulator Bit	Description
2^0	Status bit 2^0
2^1	Status bit 2^1
2^2	Status bit 2^2
2^3	Status bit 2^3
2^4	Status bit 2^4
2^5	Status bit 2^5
2^6	Status bit 2^6
2^7	Status bit 2^7
2^8	Address bit 2^0
2^9	Address bit 2^1
2^{10}	Address bit 2^2
2^{11}	Address bit 2^3
2^{12}	Address bit 2^4
2^{13}	Address bit 2^5
2^{14}	Address bit 2^6
2^{15}	Address bit 2^7

Function	Description
BMA : 13	Reads the input tags from the channel to the IOP accumulator. Because of the fixed timing in the IOP, this function must be run twice in immediate succession to get valid input tags to the IOP accumulator. The first performance of the function moves the input tags to the block multiplexer controller. The second performance of the function moves the input tags to the IOP accumulator. A BMA : 13 function following a BMA : 17 function verifies correct operation of the accumulator fanout, the intermediate output tags register, and the path back to the accumulator. Issuing a BMA : 17 function loads the output tags into the output tags register. The next BMA : 13 function reads the stored output tags back into the IOP accumulator. This function does not affect the busy and done flags. Table 4-24 shows the input tags status bits.

Table 4-24. Input Tags Status Bits

Accumulator Bit	Description
2^0	Operational-in
2^1	Address-in
2^2	Disconnect-in
2^3	Select-in
2^4	Request-in
2^5	Service-in
2^6	Data-in
2^7	Status-in
2^8	Metering-in
2^9	Mark-0-in
2^{10}	Not used (BMC-4); data streaming (BMC-5)
2^{11}	Data buffer pointer
2^{12}	Address miscompare
2^{13}	Byte count 0
2^{14}	PROM parity error
2^{15}	Bus-in parity error

Function — Description

BMA : 14 — Enters the current accumulator contents into the local memory address register. This value becomes the starting local memory address for the data transfer. This function does not affect the busy and done flags.

The block multiplexer controller maintains two local memory address registers for data chaining. Accumulator bit 2^0 addresses the registers. Data-chaining begins with the address specified by local memory address register 0 and alternates between the 0 and 1 registers. Bit 2^1 of the accumulator content is the data chaining select flag for the chosen register and, if set, selects data chaining for that register/address. Table 4-25 lists the accumulator bits for this function. Function BMA : 14 clears the interrupt condition after a data transfer.

BMA : 15 — Enters the accumulator content into the byte counter or the next byte count register. The first BMA : 15 function following a BMA : 14 enters the accumulator content into the byte counter. When a second BMA : 15 function follows immediately, the second BMA : 15 function enters the accumulator data into the next byte count register.

Table 4-25. Local Memory Address Register Bits

Accumulator Bit	Description
2^0	Local memory address register select
2^1	Data chaining flag
2^2	Local memory address 2^2
2^3	Local memory address 2^3
2^4	Local memory address 2^4
2^5	Local memory address 2^5
2^6	Local memory address 2^6
2^7	Local memory address 2^7
2^8	Local memory address 2^8
2^9	Local memory address 2^9
2^{10}	Local memory address 2^{10}
2^{11}	Local memory address 2^{11}
2^{12}	Local memory address 2^{12}
2^{13}	Local memory address 2^{13}
2^{14}	Local memory address 2^{14}
2^{15}	Local memory address 2^{15}

Function Description

The contents of the next byte count register transfer to the byte counter automatically between data segments in data chaining. Channel commands requiring no data, parameters, or status must establish a byte count of 0. The maximum count is 65,535 bytes. This function does not affect the busy and done flags.

BMA : 16 Enters the accumulator contents into the device address register. The device address register contains mode select bits and device address bits. The device address is a combination of controller address bits and peripheral device address bits. This function does not affect the busy and done flags. Table 4-26 shows the device address register bits.

Parameter mode bits - Bits 2^8 through 2^{15} of the parameter are called mode bits and are re-established during each BMA : 16 function. They operate as follows.

- Bit 2^8 is the mode bit for the skip flag. When set, this bit prohibits storing data into local memory during read data transfers.

Table 4-26. Device Address Register Bits

Accumulator Bit	Description
2^0	Address/data out 2^0
2^1	Address/data out 2^1
2^2	Address/data out 2^2
2^3	Address/data out 2^3
2^4	Address/data out 2^4
2^5	Address/data out 2^5
2^6	Address/data out 2^6
2^7	Address/data out 2^7
2^8	Skip flag
2^9	Stack status flag
2^{10}	Command-chaining mode select 2^0
2^{11}	Command-chaining mode select 2^1
2^{12}	Interrupt mode select 2^0 (BMC-4)/ Request-in interrupt enable (BMC-5)
2^{13}	Interrupt mode select 2^1 (BMC-4)/ Mode bit 2^5 (BMC-5)
2^{14}	Channel mode select 2^0
2^{15}	Channel mode select 2^1

Function Description

- Bit 2^9 is the mode bit for the stack status flag.
- Bits 2^{10} and 2^{11} select the mode in which command chaining is used. Table 4-27 lists the modes and corresponding mode bits.

Function **Description**

- On a BMC-4, parameter bits 2^{12} through 2^{13} select the interrupt mode. Table 4-28 lists the modes and corresponding bits.

- On a BMC-5, parameter bit 2^{12} allows the request-in signal to generate an IOP interrupt.

- On a BMC-5, parameter bit 2^{13} selects automatic transfer of status and address in a BMA : 3 function.

- Parameter bits 2^{14} through 2^{15} select the channel protocol. Table 4-29 lists the protocols and corresponding bits for the BMC-4. Table 4-30 lists the protocols and corresponding bits for the BMC-5.

Table 4-27. Command Chaining Mode Selection

Parameter bits 2^{11} - 2^{10}	Selection
0 0	Command chaining is disabled.
0 1	Chain if the channel-end status is detected.
1 0	Chain if the device-end status is detected.
1 1	Chain if either the channel-end status or device-end status is detected.

Table 4-28. Interrupt Mode Selection

Parameter bits 2^{13} - 2^{12}	Selection
0 0	Interrupts are disabled
0 1	An interrupt occurs on request-in
1 0	An interrupt occurs on status-in
1 1	An interrupt occurs on disconnect-in

Table 4-29. Channel Type Mode Selection for the BMC-4

Parameter bits 2^{15} - 2^{14}	Selection
0 0	Selector channel
0 1	Byte multiplexer channel
1 0	Block multiplexer channel
1 1	Reserved

Table 4-30. Channel Type Mode Selection for the BMC-5

Parameter bits 2^{15} - 2^{14}	Selection
0 0	Selector channel
0 1	Data streaming channel (4.5 Mbytes/s)
1 0	Block multiplexer channel
1 1	Data streaming channel (2 and 3 Mbytes/s)

Function	Description
BMA : 17	Enters the accumulator contents into the output tags register, allowing direct program control of the output tags for special control sequences such as diagnostics. This function does not affect the busy and done flags. Table 4-31 shows the accumulator bits for each output tag.

Table 4-31. Output Tags Register Bits

Accumulator Bits	Description
2^0	Operational-out tag
2^1	Address-out tag
2^2	Hold-out tag
2^3	Select-out tag
2^4	Command-out tag
2^5	Service-out tag
2^6	Data-out tag
2^7	Suppress-out tag
2^8	Metering-out tag
2^9	Mark 0 out
2^{10}	BMC-4: this bit is not used. BMC-5: when set, this bit transfers the devide address from the device address register directly to the bus 0 out data lines.
2^{11}	BMC-4: this bit is not used. BMC-5: when set, this bit transfers the bus-0-in data bits to the device address register.
2^{12}	When this bit is set, the clock-out tag line is also set.
2^{13}	When set, this bit prevents reporting of parity errors on the bus 0 in data lines.
2^{14}	When set, this bit forces a bus-out parity error by sending an incorrect parity bit with the byte being transferred on the bus 0 out data lines. This mode affects the transfer of a single byte only.
2^{15}	BMC-4: This bit is not used. BMC-5: When set, this bit causes a 2IW module test point to go high. Monitor the test point with an oscilloscopte to determine when the BMA : 17 function issues.

CHANNEL INTERFACE TO DISK STORAGE UNITS

Refer to the *Disk Systems Hardware Reference Manual,* CRI publication number HR-0077, for information about the DCU-4 and DCU-5 channels and functions.

5 - BUFFER MEMORY/CENTRAL MEMORY BYPASS MODE

The buffer memory channel (channel 5) and the central memory channels (channels 14 and 15) can transfer data directly between mainframe central memory or a solid-state storage device (SSD) to I/O subsystem (IOS) buffer memory. The bypass operation allows transfers in only one direction at a time. It requires channel 5 and either channel 14 (input) or channel 15 (output). The other central memory channel (input or output) continues to function normally, to or from local memory.

BYPASS MODE BITS

Two bypass mode bits, set by the buffer memory channel MOS : 16 function, control data transfers in bypass mode. Only 1 mode bit may be set at a time. A MOS : 16 function with the accumulator equal to 14_8 sets 1 mode bit and causes a link with the central memory input channel. A MOS : 16 function with the accumulator equal to 15_8 sets the other mode bit and causes a link with the central memory output channel. Any of the following actions clear the bypass mode bits:

- A MOS : 16 function with the accumulator equal to 0
- A HIA : 0 function, if the central memory input channel is linked but inactive
- A HOA : 0 function, if the central memory output channel is linked but inactive
- Completion of the bypass transfer
- A Master Clear signal

The mode bits should never be set or cleared while either of the selected channels is active or has an outstanding error condition.

RESIDUE BITS

The residue bits indicate that the previous transfer on the channel was a bypass operation. One residue bit represents the central memory input channel; the other bit represents the central memory output channel. Only 1 residue bit may be set at a time. A MOS : 10 function with accumulator bit 2^1 set reads both the bypass bits and the residue bits.

A residue bit sets when the done flag sets and one of the following conditions occurs:

- The bypass operation is complete.
- The central memory channel terminates the bypass operation.
- The bypass operation is in progress, and an uncorrectable error occurs on the central memory channel.

A residue bit clears under the following conditions:

- A MOS : 1 through MOS : 5 function (start next transfer) is issued.
- A Master Clear signal is activated.

READING BYPASS MODE AND RESIDUE BITS

A MOS : 10 function reads the bypass mode bits and the residue bits if accumulator bit 2^1 is set. When the function is complete, the accumulator bit positions have the following meanings:

Bit	Decription
2^0	The channel is currently in bypass mode and linked with the buffer-memory input channel.
2^1	The channel is currently in bypass mode and linked with the buffer-memory output channel.
2^2	The previous transfer was a bypass operation for the central-memory input channel.
2^3	The previous transfer was a bypass operation for the central-memory output channel.

BYPASS MODE OPERATION

A bypass mode transfer requires execution of the following three channel functions:

- A MOS : 16 function sets the bypass bit.
- A HIA : 4 or HOA : 5 function sends the block length to the central memory interface.
- A MOS : 4 function sends the block length to the buffer memory interface and initiates the transfer.

Send matching block length values to the central memory channel interface and the buffer memory channel interface to avoid causing block length errors during the transfer. The channels remain linked until both channels complete the transfer or until software commands terminate activity on one channel. When software terminates one channel, both channels become inactive simultaneously.

Enable interrupts on only one channel for bypass transfers. The done flags of both channels set at the end of the transfer. When the done flags set, one channel interrupts the I/O processor (IOP), the bypass mode bit clears, and the central memory channel is available for normal channel operation. If interrupts are enabled on either channel, the corresponding done flag interrupts the IOP.

A bypass mode transfer begins when the MOS : 16 function sets one of the bypass mode bits and the MOS : 4 function sends the block length to the buffer memory channel interface.

NOTE: Sending the block length to the linked central memory channel interface with an HIA : 4 or HOA : 5 function does not begin a transfer.

The bypass bit clears and the residue bit sets at the end of all transfers. A central memory to buffer memory transfer ends when one of the following conditions occur:

- The transfer reaches normal completion: all words transfer to buffer memory and both done flags set.
- A function error occurs on the buffer memory channel: the IOP issues a MOS : 0 function while the transfer is active, setting a central memory channel error bit and the busy and done flags of both channels.

- A function error occurs on the central memory input channel: the IOP issues an HIA : 0 through HIA : 5 function while the transfer is active, setting a central memory channel error bit and the busy and done flags of both channels.

- Uncorrectable data errors occur on the central memory channel: the data errors set the central memory channel error bit, and the busy and done flags of both channels.

- A Master Clear signal is sent: the Master Clear signal clears the busy and done flags in the next clock period (CP).

A buffer memory to central memory transfer ends when one of the following conditions occur:

- The transfer reaches normal completion: all words transfer to central memory and both done flags set.

- A function error occurs on the buffer memory channel: the IOP issues a MOS : 0 function while the transfer is active, forcing a function error on the central memory channel interface. The function error sets a central memory channel error bit, and the busy and done flags of both channels.

- A function error occurs on the central memory output channel: the IOP issues a HOA : 0 through HOA : 5 function while the transfer is active, forcing an error on the central memory output channel interface. The error sets a central memory channel error bit and the busy and done flags of both channels.

- An uncorrectable data error occurs on the central memory channel: an uncorrectable error forces an error on the central memory output channel interface, sets an error on both channels, clears the bypass bit, and sets the residue bit and the central memory channel error bit.

- A Master Clear signal is sent: the Master Clear signal clears the busy and done flags in the next CP.

ABORTING A BYPASS OPERATION

Perform the following steps to abort a transfer in progress:

1. Issue an MOS : 0, HIA : 0, or HOA : 0 function to clear the bypass bit and force errors on both channels. The error causes the central memory channel interface to perform a clear channel sequence.

2. Issue a function 0 to each of the previously linked interfaces to clear the error conditions forced by Step 1.

****

CAUTION

Never clear the bypass bit directly (with an MOS : 16 function) if the buffer memory and central memory channels are active. Always deactivate the channel with an error condition to clear the bypass bit. Clearing the bypass bit directly during a transfer causes an unpredictable alteration to local memory contents and causes unpredictable consequences to subsequent transfers.

****

Perform the following steps to disconnect channels that have been linked for bypass operation, but that are not yet active:

1. Issue a MOS : 16 command with the accumulator equal to 0 to clear the bypass bits.

2. Issue a HIA : 0 command to clear the central memory input channel, or an HOA : 0 command to clear the central memory output channel.

NOTE: The HIA : 0 or HOA : 0 functions do not cause a function error if the central memory channel is not active. However, the MOS : 0 command always causes a central memory channel function error if the bypass bit is set.

Perform the following steps to start a transfer from central memory to buffer memory:

1. Wait for both channels to become inactive.

2. Ensure that neither channel has an error condition.

3. Set the bypass mode bit, using a MOS : 16 function.

4. Enable or disable interrupts from the central memory channel, if desired.

5. Set the central memory starting address for the central memory channel interface.

6. Set the block length for the central memory channel interface.

7. Set the buffer memory starting address for the buffer memory channel interface.

8. Set the block length for the buffer memory channel interface to start the transfer.

9. Wait for an interrupt from channel 5 (buffer memory) or the done flag to set. If an error occurs on either channel, the buffer memory channel busy flag remains set.

This section describes I/O channel assignments. Table 6-1 through 6-4 list typical channel assignments for the I/O subsystem model C (IOS-C). Tables 6-5 through 6-8 list typical channel assignments for the I/O subsystem model D (IOS-D). Figure 6-1 shows the correspondence between I/O channels and direct memory access (DMA) ports.

IOS-C

MIOP

DMA 0 Channel 17	DMA 1 Channels 20 - 27	DMA 2 Channels 30 - 37
DMA 3 Channels 5, 51, 52	DMA 4 Channels 14, 15	DMA 5

BIOP

DMA 0 Channels 20 - 23	DMA 1 Channels 24 - 27	DMA 2 Channels 30 - 33
DMA 3 Channels 5, 51, 52	DMA 4 Channels 14, 15	DMA 5 Channels 34 - 37

DIOP

DMA 0 Channels 20 - 23	DMA 1 Channels 24 - 27	DMA 2 Channels 30 - 33
DMA 3 Channels 5, 51, 52	DMA 4 Channels 14, 15	DMA 5 Channels 34 - 37

XIOP

DMA 0 Channels 20 - 23	DMA 1 Channels 24 - 27	DMA 2 Channels 30 - 33
DMA 3 Channels 5, 51, 52	DMA 4 Channels 14, 15	DMA 5 Channels 34 - 37

IOS-D

MIOP

DMA 0 Channels 16 - 17	DMA 1 Channels 20 - 27	DMA 2 Channels 30 - 37
DMA 3 Channels 5, 51, 52	DMA 4 Channels 14, 15	DMA 5 Channels 52, 53

BIOP

DMA 0 Channels 20 - 23	DMA 1 Channels 24 - 27	DMA 2 Channels 30 - 33
DMA 3 Channels 5, 51, 52	DMA 4 Channels 14, 15	DMA 5 Channels 52, 53

DIOP

DMA 0 Channels 20 - 23	DMA 1 Channels 24 - 27	DMA 2 Channels 30 - 33
DMA 3 Channels 5, 51, 52	DMA 4 Channels 14, 15	DMA 5 Channels 52 - 53

XIOP

DMA 0 Channels 5, 51, 52	DMA 1 Channels 20 - 23	DMA 2 Channels 24 - 27
DMA 3 Channels 14 - 15	DMA 4 Channels 52 - 53	DMA 5 Channels 16 - 17

Figure 6-1. DMA Port and I/O Channel Assignments

Table 6-1. Typical IOS-C MIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXS	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 3)
6	AIA	Input from the buffer I/O processor (BIOP)
7	AOA	Output to the BIOP
10	AIB	Input from the disk I/O processor (DIOP)
11	AOB	Output to the DIOP
12	AIC	Input from the auxiliary I/O processor (XIOP)
13	AOC	Output to the XIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 4)
15	HOA	Output to central memory (100 Mbyte) (DMA 4)
16		Not used
17	EXB	Peripheral expander (DMA 0)
20	CIA	Input from the front-end interface (DMA 1)
21	COA	Output to a front-end interface (DMA 1)
22	CIB	Input from a front-end interface (DMA 1)
23	COB	Output to a front-end interface (DMA 1)
24	CIC	Input from a front-end interface (DMA 1)
25	COC	Output to a front-end interface (DMA 1)
26	CID	Input from a front-end interface (DMA 1)
27	COD	Output to a front-end interface (DMA 1)
30	CIE	Input from a front-end interface (DMA 2)
31	COE	Output from a front-end interface (DMA 2)
32	CIF	Input from a front-end interface (DMA 2)
33	COF	Output from a front-end interface (DMA 2)
34	CIG	Input from a front-end interface (DMA 2)
35	COG	Output from a front-end interface (DMA 2)
36	CIH	Input from a front-end interface (DMA 2)
37	COH	Output from a front-end interface (DMA 2)
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42	TIB	Console 1 keyboard
43	TOB	Console 1 display
44-47		Available for other consoles
50	LIA	Input from the CRI mainframe I/O channel (DMA 3)
51	LOA	Output to the CRI mainframe I/O channel (DMA 3)

Table 6-2. Typical IOS-C BIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXS	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 3)
6	AIA	Input from the MIOP
7	AOA	Output from the MIOP
10	AIB	Input from the DIOP
11	AOB	Output to the DIOP
12	AIC	Input from the XIOP
13	AOC	Output to the XIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 4)
15	HOA	Output to central memory (100 Mbyte) (DMA 4)
16-17		Not used
20	DIA	Disk storage unit 0 (DMA 0)
21	DIB	Disk storage unit 1 (DMA 0)
22	DIC	Disk storage unit 2 (DMA 0)
23	DID	Disk storage unit 3 (DMA 0)
24	DIE	Disk storage unit 4 (DMA 2)
25	DIF	Disk storage unit 5 (DMA 2)
26	DIG	Disk storage unit 6 (DMA 2)
27	DIH	Disk storage unit 7 (DMA 2)
30	DII	Disk storage unit 8 (DMA 5)
31	DIJ	Disk storage unit 9 (DMA 5)
32	DIK	Disk storage unit 10 (DMA 5)
33	DIL	Disk storage unit 11 (DMA 5)
34	DIM	Disk storage unit 12 (DMA 1)
35	DIN	Disk storage unit 13 (DMA 1)
36	DIO	Disk storage unit 14 (DMA 1)
37	DIP	Disk storage unit 15 (DMA 1)
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42-47		Available for other consoles
50	LIA	Input from the CRI mainframe I/O channel (DMA 3)
51	LOA	Output to the CRI mainframe I/O channel (DMA 3)

Table 6-3. Typical IOS-C DIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXR	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 3)
6	AIA	Input from the MIOP
7	AOA	Output to the MIOP
10	AIB	Input from the BIOP
11	AOB	Output to the BIOP
12	AIC	Input from the XIOP
13	AOC	Output to the XIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 4)
15	HOA	Output to central memory (100 Mbyte) (DMA 4)
16-17		Not used
20	DIA	Disk storage unit 0 (DMA 0)
21	DIB	Disk storage unit 1 (DMA 0)
22	DIC	Disk storage unit 2 (DMA 0)
23	DID	Disk storage unit 3 (DMA 0)
24	DIE	Disk storage unit 4 (DMA 2)
25	DIF	Disk storage unit 5 (DMA 2)
26	DIG	Disk storage unit 6 (DMA 2)
27	DIH	Disk storage unit 7 (DMA 2)
30	DII	Disk storage unit 8 (DMA 5)
31	DIJ	Disk storage unit 9 (DMA 5)
32	DIK	Disk storage unit 10 (DMA 5)
33	DIL	Disk storage unit 11 (DMA 5)
34	DIM	Disk storage unit 12 (DMA 1)
35	DIN	Disk storage unit 13 (DMA 1)
36	DIO	Disk storage unit 14 (DMA 1)
37	DIP	Disk storage unit 15 (DMA 1)
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42-47		Available for other consoles
50	LIA	Input from the CRI mainframe I/O channel (DMA 3)
51	LOA	Output to the CRI mainframe I/O channel (DMA 3)

Table 6-4. Typical IOS-C XIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXS	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 3)
6	AIA	Input from the MIOP
7	AOA	Output to the MIOP
10	AIB	Input from the BIOP
11	AOB	Output to the BIOP
12	AIC	Input from the DIOP
13	AOC	Output to the DIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 4)
15	HOA	Output to central memory (100 Mbyte) (DMA 4)
16	XIA	Input from the high-speed external (HSX) channel (100 Mbyte) (DMA 5)
17	XOA	Output from the HSX channel (100 Mbyte) (DMA 5)
20	BMA	Block multiplexer channel 0 (DMA 0)
21	BMB	Block multiplexer channel 1 (DMA 0)
22	BMC	Block multiplexer channel 2 (DMA 0)
23	BMD	Block multiplexer channel 3 (DMA 0)
24	BME	Block multiplexer channel 4 (DMA 1)
25	BMF	Block multiplexer channel 5 (DMA 1)
26	BMG	Block multiplexer channel 6 (DMA 1)
27	BMH	Block multiplexer channel 7 (DMA 1)
30	BMI	Block multiplexer channel 8 (DMA 2)
31	BMJ	Block multiplexer channel 9 (DMA 2)
32	BMK	Block multiplexer channel 10 (DMA 2)
33	BML	Block multiplexer channel 11 (DMA 2)
34-37		Not used
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42-47		Available for other consoles
50	LIA	Input from the CRI mainframe I/O channel (DMA 3)
51	LOA	Output to the CRI mainframe I/O channel (DMA 3)

Table 6-5. Typical IOS-D MIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXS	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 3)
6	AIA	Input from the BIOP
7	AOA	Output to the BIOP
10	AIB	Input from the DIOP
11	AOB	Output to the DIOP
12	AIC	Input from the XIOP
13	AOC	Output to the XIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 4)
15	HOA	Output to central memory (100 Mbyte) (DMA 4)
16	CIA	Input to the VME operator workstation (DMA 0)
17	COA	Output to the VME operator workstation (DMA 0)
20	CIB	Input from the front-end interface (DMA 1)
21	COB	Output to the front-end interface (DMA 1)
22	CIC	Input from the front-end interface (DMA 1)
23	COC	Output to the front-end interface (DMA 1)
24	CID	Input from the front-end interface (DMA 1)
25	COD	Output to the front-end interface (DMA 1)
26	CIE	Input from the front-end interface (DMA 1)
27	COE	Output to the front-end interface (DMA 1)
30	CIF	Input from the front-end interface (DMA 2)
31	COF	Output to the front-end interface (DMA 2)
32	CIG	Input from the front-end interface (DMA 2)
33	COG	Output to the front-end interface (DMA 2)
34	CI	Input from the front-end interface (DMA 2)
35	CO	Output to the front-end interface (DMA 2)
36	CI	Input from the front-end interface (DMA 2)
37	CO	Output to the front-end interface (DMA 2)
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42	TIB	Console 1 keyboard
43	TOB	Console 1 display
44-47		Available for other consoles
50	LIA	Input from CRI mainframe I/O channel (DMA 3)
51	LOA	Output to CRI mainframe I/O channel (DMA 3)
52	HIB	Input from central memory (100 Mbyte) (DMA 5)
53	HOB	Output to central memory (100 Mbyte) (DMA 5)

Table 6-6. Typical IOS-D BIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXS	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 3)
6	AIA	Input from the MIOP
7	AOA	Output from the the MIOP
10	AIB	Input from the DIOP
11	AOB	Output to the DIOP
12	AIC	Input from the XIOP
13	AOC	Output to the XIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 4)
15	HOA	Output to central memory (100 Mbyte) (DMA 4)
16-17		Not used
20	DIA	Disk storage unit 0 (DMA 0)
21	DIB	Disk storage unit 1 (DMA 0)
22	DIC	Disk storage unit 2 (DMA 0)
23	DID	Disk storage unit 3 (DMA 0)
24	DIE	Disk storage unit 4 (DMA 1)
25	DIF	Disk storage unit 5 (DMA 1)
26	DIG	Disk storage unit 6 (DMA 1)
27	DIH	Disk storage unit 7 (DMA 1)
30	DII	Disk storage unit 8 (DMA 2)
31	DIJ	Disk storage unit 9 (DMA 2)
32	DIK	Disk storage unit 10 (DMA 2)
33-37		Not used
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42-47		Available for other consoles
50	LIA	Input from CRI mainframe I/O channel (DMA 3)
51	LOA	Output to CRI mainframe I/O channel (DMA 3)
52	HIB	Input from central memory (100 Mbyte) (DMA 5)
53	HOB	Output to central memory (100 Mbyte) (DMA 5)

Table 6-7. Typical IOS-D DIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXR	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 3)
6	AIA	Input from the MIOP
7	AOA	Output to the MIOP
10	AIB	Input from the BIOP
11	AOB	Output to the BIOP
12	AIC	Input from the XIOP
13	AOC	Output to the XIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 4)
15	HOA	Output to central memory (100 Mbyte) (DMA 4)
16-17		Not used
20	DIA	Disk storage unit 0 (DMA 0)
21	DIB	Disk storage unit 1 (DMA 0)
22	DIC	Disk storage unit 2 (DMA 0)
23	DID	Disk storage unit 3 (DMA 0)
24	DIE	Disk storage unit 4 (DMA 1)
25	DIF	Disk storage unit 5 (DMA 1)
26	DIG	Disk storage unit 6 (DMA 1)
27	DIH	Disk storage unit 7 (DMA 1)
30	DII	Disk storage unit 8 (DMA 2)
31	DIJ	Disk storage unit 9 (DMA 2)
32	DIK	Disk storage unit 10 (DMA 2)
33	DIL	Disk storage unit 11 (DMA 2)
34-37		Not used
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42-47		Available for other consoles
50	LIA	Input from the CRI mainframe I/O channel (DMA 3)
51	LOA	Output to the CRI mainframe I/O channel (DMA 3)
52	HIB	Input from central memory (100 Mbyte) (DMA 5)
53	HOB	Output to central memory (100 Mbyte) (DMA 5)

Table 6-8. Typical IOS-D XIOP Channel Assignments

Channel (octal)	Mnemonic	Description
0	IOR	Interrupt request
1	PFR	Program fetch request
2	PXS	Program exit stack
3	LME	Local memory error
4	RTC	Real-time clock
5	MOS	Buffer memory interface (DMA 0)
6	AIA	Input from the MIOP
7	AOA	Output to the MIOP
10	AIB	Input from the BIOP
11	AOB	Output to the BIOP
12	AIC	Input from the DIOP
13	AOC	Output to the DIOP
14	HIA	Input from central memory (100 Mbyte) (DMA 3)
15	HOA	Output to central memory (100 Mbyte) (DMA 3)
16	XIA	Input from an HSX channel (100 Mbyte) (DMA 5)
17	XOA	Output to an HSX channel (100 Mbyte) (DMA 5)
20	BMA	Block multiplexer channel 0 (DMA 1)
21	BMB	Block multiplexer channel 1 (DMA 1)
22	BMC	Block multiplexer channel 2 (DMA 1)
23	BMD	Block multiplexer channel 3 (DMA 1)
24	BME	Block multiplexer channel 4 (DMA 2)
25	BMF	Block multiplexer channel 5 (DMA 2)
26	BMG	Block multiplexer channel 6 (DMA 2)
27	BMH	Block multiplexer channel 7 (DMA 2)
30-37		Not used
40	TIA	Console 0 keyboard
41	TOA	Console 0 display
42-47		Available for other consoles
50	LIA	Input from the CRI mainframe I/O channel (DMA 0)
51	LOA	Output to the CRI mainframe I/O channel (DMA 0)
52	HIB	Input from central memory (100 Mbyte) (DMA 4)
53	HOB	Output to central memory (100 Mbyte) (DMA 4)

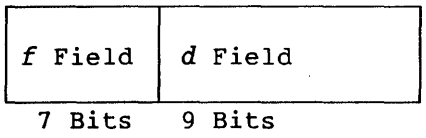
7 - IOP INSTRUCTION SET

This section describes the I/O processor instruction set. The instruction set enables a programmer to write programs that load, shift, add, subtract, increment, and decrement values, calculate a logical product, test for busy and done flags, move around in a program (relative and absolute branching), and enable and disable interrupts.

INSTRUCTION FORMAT

Each I/O processor (IOP) instruction occupies 1 parcel (16 bits) or 2 parcels (32 bits) in local memory. A 1-parcel instruction consists of a function code (*f*) field and a designator (*d*) field. A 2-parcel instruction consists of the *f* and *d* fields and a constant (*k*) field. Refer to Figure 5-1.

1-parcel Instruction:



2-parcel Instruction:

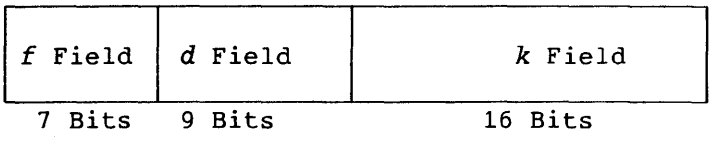


Figure 7-1. 1- and 2-parcel IOP Instruction Formats

The 7-bit *f* field specifies the machine instruction to be issued. The 9-bit *d* field performs one of the following functions:

- Points to a specific operand register
- Points to a specific I/O channel
- Specifies the amount of data displacement in a shift instruction
- Specifies the amount of displacement forward or backward in program code for a branch instruction

- Supplies an operand value

The *d* field is always treated as a 9-bit positive integer. Some instructions use the d field as a constant for addition and subtraction. Some branch instructions use the d field as a forward or a backward displacement from the current program location.

Two-parcel instructions use the program parcel immediately following the instruction as a 16-bit constant field, designated *k*.

INSTRUCTION ISSUE CONFLICTS

Accumulator, register, and branch issue conflicts may delay instruction issue.

ACCUMULATOR CONFLICTS

Accumulator conflicts arise when two subsequent instructions both need the accumulator at the same time. Instructions that use the accumulator cannot issue unless the previous instruction will be done using the accumulator when the new instruction requires the accumulator. Three types of instructions cause accumulator conflicts:

- Instructions that require the accumulator to be free at issue
- Instructions that require the accumulator to be free 1 CP after issue
- Instructions that require the accumulator to be free 2 CPs after issue

The following instructions require the accumulator to be free:

Instructions	Requirement
22, 23, 25, 30-37, 40-43, 62, 63, and 65	Free in 2 CPs
4-7, 11-13, 15-17, 20, 21, 24, 26, 27, 44-47, 51-53, 55, 60, 61, 64, 66, 67, and 140-177	Free in 1 CP
10, 14, 50, 54, 56, and 57	Free at issue

Instructions that use the accumulator as a destination force the accumulator to be busy when they issue. The exception is instructions that complete in 1 CP, which do not force the accumulator to be busy. The accumulator remains busy until the CP noted in the instruction description.

REGISTER CONFLICTS

Register conflicts arise for the destination pointer (DP) and B registers, the DP to register pointer (RP) data path, and the DP register equal to RP register.

The DP register contains a 9-bit pointer that selects one of the operand registers to receive the contents of the accumulator. When the DP register is full, a write to an operand register must occur. A register conflict occurs if the DP register is full and an instruction to write to an operand register is ready to issue. The instruction cannot issue until the write operation in progress is complete.

Instructions that reserve the B register for storing results cause B register conflicts. Other instructions that use the B register cannot issue until the results are stored and the B register reservation is released.

A DP-to-RP path conflict can arise during a write operation to an operand register when the address in the DP register transfers into the RP register. The DP-to-RP path is busy for 1 CP. Instructions requiring a read operation from the operand registers cannot issue during this CP.

The following instructions may cause DP, B, or DP-to-RP path register conflicts:

Instructions	Conflict
11, 15, 24-27, 51, 64-67	DP full
11, 15, 42-67, 160-177	B reservation
20-37, 60-67, 74-77, 120-137	DP-to-RP path

Logical product instructions 11, 15, and 51 can cause both DP full and B reservation conflicts because their issue conditions require the accumulator to be free in 1 CP. If the DP register is full or the B register is reserved by another instruction, the accumulator may be modified before data is stored in either the DP or B registers. Because the logical product instructions use the accumulator and run in 1 CP, the wrong value may be stored in the registers.

A register conflict also occurs when the DP register contents are the same as the RP register contents. This condition only occurs with instructions that require the accumulator to be free in 2 CPs. The condition is not directly related to instruction issue.

The DP register contents are the same as the RP register contents when an instruction requires the operand register for an operand, but the same register is currently a destination for a register store operation. To ensure the correct value is used as the operand, the read operation must wait until the store operation takes place. The read operation must also use the *read around* path that is available. Data on this path is available in the CP that it is being written. For all instructions except branches, the use of the read around path requires an instruction issue delay of 1 CP. The register is then written the following CP and the data is available through the read around path.

This same conflict cannot occur with instructions that require the accumulator to be free in 1 CP because instruction issue takes place in the same CP that the DP-to-RP path conflict would be present.

BRANCH ISSUE CONFLICTS

The following conditions delay issue of branch instructions:

- The next instruction parcel must be available for any branch to issue. The program address (P) register must increment when the branch issues so that the return address can be stored in the program exit stack.
- The carry flag must not be reserved at issue of conditional branches that test the carry flag.
- The accumulator must be free the last CP of conditional branches that test the accumulator.
- A relative branch (070 through 073, 100 through 117) cannot issue unless the adder functional unit will be available in the CP after issue. Issue of these instructions is delayed if a delayed addition instruction (such as 22 or 32) must use the adder functional unit in the CP after issue.

INSTRUCTIONS

Table 7-1 lists each instruction; the remainder of this section describes each instruction in detail.

INSTRUCTION ABBREVIATIONS, CONVENTIONS, AND SYMBOLS

Mnemonics for each instruction are A Programming Machine Language (APML) statements representing the individual operations performed. In addition, Table 7-1 and the instruction descriptions use the following abbreviations:

Abbreviation	Definition
A	IOP accumulator
C	Carry flag
CIP	Current instruction parcel register
CP	Clock period
d	The *d* field of a central processing unit (CPU) instruction
DP	Destination pointer register
E	E register of the program exit stack
f	The *f* field of a CPU instruction
IOP	I/O processor
k	The *k* field of an IOP instruction
MA	Memory address register
m, n	Variable instruction cycle times
I/O	Input/output
P	Program address register
RP	Register pointer register

Table 7-1 and the instruction descriptions use the following symbols:

Symbol	Definition
>	Shift right
<	Shift left
>>	Shift right circular
<<	Shift left circular
&	Logical product
#	Not equal to

Table 7-1 and the instruction descriptions use the following conventions:

- *dd* indicates the contents of the operand register specified by the instruction *d* field.
- (*dd*) indicates the contents of the memory location specified by *dd*.
- *iod* indicates a three-character channel mnemonic, such as IOR or PXS.
- B indicates the contents of the B register.
- (B) indicates the contents of the operand register specified by the B register.
- A - (dash) after a CP *n* means the same conditions hold for that CP as for the previous CP.

Table 7-1. IOP Instructions

IOP Instruction	APML	Description
000	PASS	No operation
001	EXIT	Exits from subroutine
002	I = 0	Disables system interrupts
003	I = 1	Enables system interrupts
004	A = A > *d*	Rights shift C and A by *d* places, end off
005	A = A < *d*	Lefts shift C and A by *d* places, end off
006	A = A >> *d*	Rights shift C and A by *d* places, circular
007	A = A << *d*	Lefts shift C and A by *d* places, circular
010	A = *d*	Transmits *d* to A
011	A = A & *d*	Forms the logical product of A and *d*, result to A
012	A = A + *d*	Adds *d* to A
013	A = A - *d*	Subtracts *d* from A
014	A = *k*	Transmits *k* to A
015	A = A & *k*	Transmits logical product of A and *k* to A
016	A = A + *k*	Adds *k* to A
017	A = A - *k*	Subtracts *k* from A
020	A = *dd*	Transmits operand register *d* to A
021	A = A & *dd*	Forms logical product of A and operand register *d*, result to A
022	A = A + *dd*	Adds operand register *d* to A
023	A = A - *dd*	Subtracts operand register *d* from A
024	*dd* = A	Transmits A to register *d*
025	*dd* = A + *dd*	Adds operand register *d* to A, result to operand register *d*

Table 7-1. IOP Instructions (continued)

IOP Instruction	APML	Description
026	*dd* = *dd* + 1	Transmits register *d* to A, adds 1, result to operand register *d*
027	*dd* = *dd* - 1	Transmits register *d* to A, subtracts 1, result to operand register *d*
030	A = (*dd*)	Transmits contents of memory addressed by register *d* to A
031	A = A & (*dd*)	Forms logical product of A and contents of memory addressed by register *d*, result to A
032	A = A + (*dd*)	Adds contents of memory addressed by register *d* to A, result to A
033	A = A - (*dd*)	Subtracts contents of memory addressed by register *d* from A, result to A
034	(*dd*) = A	Transmits A to memory addressed by register *d*
035	(*dd*) = A + (*dd*)	Adds memory addressed by register *d* to A, result to same memory location
036	(*dd*) = (*dd*) + 1	Transmits memory addressed by register *d* to A, add 1, result to same memory location
037	(*dd*) = (*dd*) - 1	Transmits memory addressed by register *d* to A, subtract 1, result to same memory location
040	C = 1, *iod* = DN	Sets carry equal to channel *d* done
041	C = 1, *iod* = BZ	Sets carry equal to channel *d* busy
042	C = 1, IOB = DN	Sets carry equal to channel B done
043	C = 1, IOB = BZ	Sets carry equal to channel B busy
044	A = A > B	Right shifts C and A by B places, end off
045	A = A < B	Left shifts C and A by B places, end off

Table 7-1. IOP Instructions (continued)

IOP Instruction	APML	Description
046	A = A >> B	Right shifts C and A by B places, circular
047	A = A << B	Left shifts C and A by B places, circular
050	A = B	Transmits B to A
051	A = A & B	Forms logical product of A and B, result to A
052	A = A + B	Adds B to A, result to A
053	A = A - B	Subtracts B from A, result to A
054	B = A	Transmits A to B
055	B = A + B	Adds B to A, result to B
056	B = B + 1	Transmits B to A, adds 1, result to B
057	B = B - 1	Transmits B to A, subtracts 1, result to B
060	A = (B)	Transmits operand register B to A
061	A = A & (B)	Forms logical product of A and operand register B, result to A
062	A = A + (B)	Adds operand register B to A, result to A
063	A = A - (B)	Subtracts operand register B from A, result to A
064	(B) = A	Transmits A to operand register B
065	(B) = A + (B)	Adds operand register B to A, result to operand register B
066	(B) = (B) + 1	Transmits operand register B to A, adds 1, result to operand register B
067	(B) = (B) - 1	Transmits operand register B to A, subtracts 1, result to operand register B
070	P = P + *d*	Jumps to P + *d*

Table 7-1. IOP Instructions (continued)

IOP Instruction	APML	Description
071	P = P - *d*	Jumps to P - *d*
072	P = P + *d*	Jumps to P + *d*, returns on completion
073	P = P - *d*	Jumps to P - *d*, returns on completion
074	P = *dd*	Jumps to address in operand register *d*
075	P = *dd* + *k*	Jumps to sum of *k* and operand register *d*
076	R = *dd*	Jumps to address in operand register *d*, returns on completion
077	R = *dd* + *k*	Jumps to sum of *k* and operand register *d*, returns on completion
100	P = P + *d*, C = 0	Jumps to P + *d* if carry = 0
101	P = P + *d*, C # 0	Jumps to P + *d* if carry # 0
102	P = P + *d*, A = 0	Jumps to P + *d* if A = 0
103	P = P + *d*, A # 0	Jumps to P + *d* if A # 0
104	P = P - *d*, C = 0	Jumps to P - *d* if carry = 0
105	P = P - *d*, C # 0	Jumps to P - *d* if carry # 0
106	P = P - *d*, A = 0	Jumps to P - *d* if A = 0
107	P = P - *d*, A # 0	Jumps to P - *d* if A # 0
110	R = P + *d*, C = 0	Jumps to P + *d* if carry = 0, returns on completion
111	R = P + *d*, C # 0	Jumps to P + *d* if carry # 0, returns on completion
112	R = P + *d*, A = 0	Jumps to P + *d* if A = 0, returns on completion
113	R = P + *d*, A # 0	Jumps to P + *d* if A # 0, returns on completion

Table 7-1. IOP Instructions (continued)

IOP Instruction	APML	Description
114	R = P - *d*, C = 0	Jumps to P - *d* if carry = 0, returns on completion
115	R = P - *d*, C # 0	Jumps to P - *d* if carry # 0, returns on completion
116	R = P - *d*, A = 0	Jumps to P - *d* if A = 0, returns on completion
117	R = P - *d*, A # 0	Jumps to P - *d* if A # 0, returns on completion
120	P = *dd*, C = 0	Jumps to address in operand register *d* if carry = 0
121	P = *dd*, C # 0	Jumps to address in operand register *d* if carry = 0
122	P = *dd*, A = 0	Jumps to address in operand register *d* if A = 0
123	P = *dd*, A # 0	Jumps to address in operand register *d* if A # 0
124	P = *dd* + *k*, C = 0	Jumps to address in operand register *d* + *k* if carry = 0
125	P = *dd* + *k*, C # 0	Jumps to address in operand register *d* + *k* if carry # 0
126	P = *dd* + *k*, A = 0	Jumps to address in operand register *d* + *k* if A = 0
127	P = *dd* + *k*, A # 0	Jumps to address in operand register *d* + *k* if A = 0
130	R = *dd*, C = 0	Jumps to address in operand register *d* if carry = 0, returns on completion
131	R = *dd*, C # 0	Jumps to address in operand register *d* if carry # 0, returns on completion
132	R = *dd*, A = 0	Jumps to address in operand register *d* if A = 0, returns on completion

Table 7-1. IOP Instructions (continued)

IOP Instruction	APML	Description
133	R = *dd*, A # 0	Jumps to address in operand register *d* if A # 0, returns on completion
134	R = *dd* + *k*, C = 0	Jumps to address in operand register *d* + *k* if carry = 0, returns on completion
135	R = *dd* + *k*, C # 0	Jumps to address in operand register *d* + *k* if carry # 0, returns on completion
136	R = *dd* + *k*, A = 0	Jumps to address in operand register *d* + *k* if A = 0, returns on completion
137	R = *dd* + *k*, A # 0	Jumps to address in operand register *d* + *k* if A # 0
140	*iod* : 0	Channel *d* function 0
141	*iod* : 1	Channel *d* function 1
142	*iod* : 2	Channel *d* function 2
143	*iod* : 3	Channel *d* function 3
144	*iod* : 4	Channel *d* function 4
145	*iod* : 5	Channel *d* function 5
146	*iod* : 6	Channel *d* function 6
147	*iod* : 7	Channel *d* function 7
150	*iod* : 10	Channel *d* function 10
151	*iod* : 11	Channel *d* function 11
152	*iod* : 12	Channel *d* function 12
153	*iod* : 13	Channel *d* function 13
154	*iod* : 14	Channel *d* function 14
155	*iod* : 15	Channel *d* function 15
156	*iod* : 16	Channel *d* function 16
157	*iod* : 17	Channel *d* function 17
160	IOB : 0	Channel B function 0
161	IOB : 1	Channel B function 1
162	IOB : 2	Channel B function 2
163	IOB : 3	Channel B function 3

Table 7-1. IOP Instructions (continued)

IOP Instruction	APML	Description
164	IOB : 4	Channel B function 4
165	IOB : 5	Channel B function 5
166	IOB : 6	Channel B function 6
167	IOB : 7	Channel B function 7
170	IOB : 10	Channel B function 10
171	IOB : 11	Channel B function 11
172	IOB : 12	Channel B function 12
173	IOB : 13	Channel B function 13
174	IOB : 14	Channel B function 14
175	IOB : 15	Channel B function 15
176	IOB : 16	Channel B function 16
177	IOB : 17	Channel B function 17

Instruction 000

IOP	AMPL	Description
000	PASS	No operation

Hold Issue Conditions

There are none.

Instruction Timing

Cycle	Action
CP 0	Issue the instruction.

Description

Instruction 000 performs no operation. This instruction's primary function is to fill program fields with null operations.

Instruction 001

IOP	AMPL	Description
001	EXIT	Exits from subroutine

Hold Issue Conditions

There are none.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Decrements the E register.
CP 2	Transmits the program exit stack data to the P register.

Description

Instruction 001 terminates the current program sequence and returns to the sequence that was suspended by calling this subprogram. The current P-register value is discarded. The program exit stack location currently indicated by the E register provides the beginning address for the reinitiated sequence. The value of the E register then decrements by 1. If the value of the E register was previously 0, decrementing is blocked and the exit stack boundary flag is set. The exit stack boundary flag causes an interruption of the program sequence for restructuring the content of the program exit stack.

If the exit instruction follows a modification of the program exit stack or the E pointer, at least 4 CPs must elapse between the last modification and the exit instruction.

Instruction 002

IOP	AMPL	Description
002	I = 0	Disables system interrupts

Hold Issue Conditions

There are none.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Clears the system interrupt enable flag.

Description

Instruction 002 clears the system interrupt enable flag.

Instruction 003

IOP	AMPL	Description
003	I = 1	Enables system interrupts

Hold Issue Conditions

There are none.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Issues the next instruction.
CP 2	Sets the system interrupt enable flag.

Description

Instruction 003 sets the system interrupt enable flag.

The IOP does not enable interrupts until after the next instruction is issued. This delay allows the interrupt program to re-enable the interrupt mode and then exit to the interrupted program.

If instruction 002 follows instruction 003, instruction 002 takes precedence and system interrupts are disabled.

Instruction 004

IOP	AMPL	Description
004	A = A > *d*	Right shifts C and A by *d* places, end off

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits *d* to the addend register.
CP 2	Transmits the accumulator data and inverted *d* to the shifter; performs the shift. The addend register is available this CP.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 004 shifts the contents of the accumulator and the carry flag to the right by *d* bit positions. The carry flag is the high-order bit and is located to the left of the accumulator contents for this operation. The shift instruction enters 0's in the carry flag and propagates them to the right as the shift progresses. No shift is performed if the shift count is 0. A shift count greater than 16_{10} clears the accumulator and carry flag.

The instruction inverts the low-order 5 bits of the addend register to provide the shift count. The other addend register bits are ignored.

Instruction 005

IOP	AMPL	Description
004	A = A < *d*	Left shifts C and A by *d* places, end off

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits *d* to the addend register.
CP 2	Transmits the accumulator data and *d* to the shifter; performs the shift. The addend register is available this CP.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 005 shifts the content of the accumulator and the carry flag to the left by *d* bit positions. The carry flag is the high-order bit and is located to the left of the accumulator for this operation. The shift instruction enters 0's in the low-order bit positions of the accumulator and propagates them to the left as the shift progresses. Bits shifted from the carry flag are discarded. No shift is performed if the shift count is 0. The instruction clears the accumulator and carry flag if the shift count is greater than or equal to 17_{10}.

The low-order 5 bits of the addend register provide the shift count. The other addend register bits are ignored.

Instruction 006

IOP	AMPL	Description
006	A = A >> *d*	Right shifts C and A by *d* places, circular

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits *d* to the addend register.
CP 2	Transmits the accumulator data and inverted *d* to the shifter. Perform the shift. The addend register is available this CP.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 006 shifts the contents of the accumulator and the associated carry flag to the right in a circular mode by *d* bit positions. The carry flag is the high-order bit and is located to the left of the accumulator for this operation. This instruction returns the bits shifted from the right end of the accumulator to the carry flag. No bits are discarded. No shift is performed if the shift count is 0.

The instruction inverts the low-order 5 bits of the addend register to determine the shift count. The other addend register bits are ignored.

Instruction 007

IOP	AMPL	Description
007	A = A << *d*	Right shifts C and A by *d* places, circular

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits *d* to the addend register.
CP 2	Transmits the accumulator data and *d* to the shifter; performs the shift. The addend register is available this CP.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 007 shifts the contents of the accumulator and the associated carry flag to the left in a circular mode by *d* bit positions. The carry flag is the high-order bit and is located to the left of the accumulator for this operation. This instruction returns bits shifted from the carry flag to the low-order bit position in the accumulator. No bits are discarded. No shift is performed if the shift count is 0.

The low-order 5 bits of the addend register provide the shift count. The other addend register bits are ignored.

Instruction 010

IOP	AMPL	Description
010	A = *d*	Transmits *d* to A

Hold Issue Conditions

The accumulator is free.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits *d* to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 010 enters the *d* field into the low-order 9 bits of the accumulator and clears the carry flag. The high-order accumulator bits are 0.

Instruction 011

IOP	AMPL	Description
011	A = A & *d*	Forms logical product of A and *d*, result to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

The DP register is not full.

Instruction Timing

Cycle	Action
CP 0	Issue the instruction.
CP 1	Transmit *d* and the accumulator contents to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 011 forms the bit-by-bit logical product of the previous accumulator contents and the *d* field, and places the result in the accumulator. The *d* field is treated as a 9-bit positive integer. Circuitry at the input of the accumulator forms the logical product. This instruction clears the carry flag.

Instruction 012

IOP	AMPL	Description
012	A = A + *d*	Adds A to *d*

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits *d* to the addend register.
CP 2	Transmits data to the adder; performs addition.
CP 3	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 012 adds the *d* field to the previous accumulator contents in 16-bit, two's-complement mode and places the result in the accumulator. The addition treats the *d* field as a 9-bit positive integer. This instruction complements the carry flag if the addition process propagates a carry bit.

Instruction 013

IOP	AMPL	Description
013	A = A - *d*	Subtracts *d* from A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits *d* to the addend register.
CP 2	Transmits data to the adder; performs subtraction.
CP 3	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 013 subtracts the *d* field from the previous accumulator contents in a 16-bit, two's-complement mode and places the result in the accumulator. This operation treats the *d* field as a 9-bit positive integer. The instruction performs subtraction by complementing the contents of the addend register and adding the result to the previous accumulator contents. A 1 is then added to the result. The instruction complements the carry flag if either addition process propagates a carry bit.

Instruction 014

IOP	AMPL	Description
014	A = *k*	Transmits *k* to A

Hold Issue Conditions

The accumulator is free.

Instruction Timing

Cycle	Action
CP 0	Reads the parcel containing the constant (*k*) field.
CP 1	Issues the instruction. Instruction issue is delayed if the next parcel is not available from the instruction stack.
CP 2	Transmits *k* data to accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 014 enters a 16-bit constant in the accumulator and clears the carry flag. The next sequential parcel in the program field provides the constant.

Instruction 015

IOP	AMPL	Description
015	A = A & *k*	Forms logical product of A and *k*, result to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

The DP register is not full.

Instruction Timing

Cycle	Action
CP 0	Reads the parcel containing the constant (*k*) field.
CP 1	Issues the instruction. Issue is delayed if the next parcel is not available from the instruction stack.
CP 2	Transmits *k* to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 015 forms the bit-by-bit logical product of the previous accumulator contents and a 16-bit constant, and places the result in the accumulator. It then clears the carry flag. The next sequential parcel in the program field provides the constant. Circuitry at the input of the accumulator forms the logical product.

Instruction 016

IOP	AMPL	Description
016	A = A + *k*	Adds *k* to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Reads the parcel containing the constant (*k*) field.
CP 1	Issues the instruction. Issue is delayed if the next parcel is not available from the instruction stack.
CP 2	Transmits the *k* field contents to the addend register.
CP 3	Transmits data to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 016 adds a 16-bit constant to the previous accumulator contents in a two's complement mode and places the result in the accumulator. The next sequential parcel in the program field provides the constant. The instruction complements the carry flag if the addition process propagates a carry bit.

Instruction 017

IOP	AMPL	Description
017	A = A - *k*	Subtracts *k* from A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Reads the parcel containing the constant (*k*) field.
CP 1	Issues the instruction. Issue is delayed if the next parcel is not available from the instruction stack.
CP 2	Transmits the *k* field to the addend register.
CP 3	Transmits data to the adder; performs subtraction.
CP 4	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 017 subtracts a 16-bit constant from the previous accumulator contents in a two's complement mode and places the result in the accumulator. The next sequential parcel in the program field provides the constant. Subtraction is performed by complementing the constant and adding the result to the accumulator contents. A 1 is then added to the result. The instruction complements the carry flag if either addition process propagates a carry bit.

NOTE: Instruction 016 performs a function equivalent to 017 by using a different constant. The list includes instruction 017 to make complete translational pattern.

Instruction 020

IOP	AMPL	Description
020	A = *dd*	Transmits operand register *d* to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number specified by *d* to the RP register.
CP 2	Transmits the operand register contents to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 020 enters the contents of the operand register specified by *d* in the accumulator and clears the carry flag.

Instruction 021

IOP	AMPL	Description
021	A = A & *dd*	Forms the logical product of A and operand register *d*, results to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register data to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 021 forms the bit-by-bit logical product of the previous accumulator contents and the contents of the operand register *d* and places the result in the accumulator. It then clears the carry flag.

The input of the accumulator forms the logical product of the previous accumulator contents. The logical product function requires no additional time.

Instruction 022

IOP	AMPL	Description
022	A = A + *dd*	Adds operand register *d* to A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register data to the addend register. Transmission is delayed if the DP register was full and DP = RP.
CP 3	Transmits data from the accumulator and the addend register to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 022 adds the contents of operand register *d* to the previous accumulator contents. Addition is performed in the twos complement mode. The instruction complements the carry flag if the addition process propagates a carry bit.

Instruction 023

IOP	AMPL	Description
023	A = A - *dd*	Subtracts operand register *d* from A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register data to the addend register. Transmission is delayed if the DP register is full and DP = RP.
CP 3	Transmits data from the accumulator and the addend register to the adder; perform subtraction.
CP 4	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 023 subtracts the contents of operand register *d* from the previous accumulator contents. Subtraction is performed by complementing the subtrahend and adding the result to the accumulator contents in one's complement mode. A 1 is then added to the result. The instruction complements the carry flag if the addition process propagates a carry bit.

Instruction 024

IOP	AMPL	Description
024	*dd* = A	Transmits A to operand register *d*

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The DP register is not full.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP and DP registers.
CP 2	Transmits the accumulator data to the operand register. The accumulator is free for subsequent instructions.

Description

Instruction 024 stores the accumulator contents in operand register *d*.

Instruction 025

IOP	AMPL	Description
025	*dd* = A + *dd*	Adds operand register *d* to A, result to operand register A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

The DP register is not full.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the contents of *d* to the RP and DP registers. Indicates that the DP register is full.
CP 2	Transmits the contents of operand register *d* to the addend register. Discards the RP register pointer.
CP 3	Transmits data to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. Transmits DP register data to the RP register, re-entering the same pointer that was discarded in CP 2. The accumulator is free for subsequent instructions. Indicates that the DP register is empty.
CP 5	Transmits accumulator data to the operand register.

Description

Instruction 025 adds the contents of the operand register specified by *d* to the previous accumulator contents. Addition is performed in two's complement mode. The instruction complements the carry flag if the addition process propagates a carry bit. It then replaces the contents of operand register *d* with the new accumulator contents.

Instruction 026

IOP	AMPL	Description
026	*dd* = *dd* + 1	Transmit *d* to A, add 1, result to operand register *d*

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

The DP register is not full.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP and DP registers. Indicates that the DP register is full.
CP 2	Transmit operand register data to the addend register. Enter a +1 in the accumulator. Clears the carry flag. Discards the RP register pointer.
CP 3	Transmits data to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. Transmits the DP register data to the RP register, reentering the same pointer discarded in CP 2. The accumulator is free for subsequent instructions. Indicates that the DP register is empty.
CP 5	Transmits a copy of the accumulator data to the operand register.

Description

Instruction 026 replaces the contents of operand register *d* with the previous contents increased by 1. The instruction clears the carry flag at the beginning of this operation and enters a 1 in the accumulator. The contents of the operand register enter the addend register and are added to the accumulator contents in two's complement mode. The instruction sets the carry flag if the addition process propagates a carry bit. The result is returned to the accumulator and then to the operand register.

Instruction 027

IOP	AMPL	Description
027	*dd* = *dd* - 1	Transmits *d* to A, subtract 1, result to operand register *d*

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

The DP register is not full.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP and DP registers. Indicates that the DP register is full.
CP 2	Transmits operand register data to the addend register. Enters a -1 in the accumulator. Discards the RP register pointer. Clears the carry flag.
CP 3	Transmits data to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. Transmits DP register data to the RP register, reentering the pointer discarded in CP 2. The accumulator is free for subsequent instructions. Indicates that the DP register is empty.
CP 5	Transmits a copy of the accumulator data to the operand register.

Description

Instruction 027 replaces the contents of operand register *d* with the previous contents decreased by 1. The result is left in the accumulator and in the operand register. The instruction initially clears the carry flag and sets all accumulator bits. The operand register contents are entered in the addend register and then added to the accumulator contents. The instruction sets the carry flag if the addition process propagates a carry bit. The result goes to the operand register.

Instruction 030

IOP	AMPL	Description
030	A = (*dd*)	Transmits contents of memory addressed by register *d* to A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issue the instruction.
CP 1	Transmit register number *d* to the RP register.
CP 2	Transmit operand register data to the MA register. Transmission is delayed if the DP register was full and DP = RP.
CP 3	Transmit MA register data to the bank address registers. Send a read request to local memory. This operation repeats until the acceptance signal is received.
CP 4	The IOP computation section receives an acceptance signal from local memory.
CP 5-6	The IOP computation section waits for data from local memory.
CP 7	Transmit local memory data to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 030 enters the contents of a local memory location in the accumulator. The operand register specified by *d* supplies the local memory address. The instruction clears the carry flag.

Instruction 031

IOP	AMPL	Description
031	A = A & (*dd*)	Forms the logical product of A and contents of memory addressed by register *d*, result to A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits operand register data to the MA register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits the MA register data to the bank address registers. Send the Read Request signal to local memory. The function of CP 3 repeats until the acceptance signal is received.
CP 4	The IOP computation section receives the acceptance signal from local memory.
CP 5-6	The IOP computation section waits for data from local memory.
CP 7	Transmits the local memory data to the accumulator. Circuitry at the input of the accumulator performs the logical product of memory data and the accumulator contents. The logical product function does not require additional time. The accumulator is free for subsequent instructions.

Description

Instruction 031 forms the bit-by-bit logical product of the previous accumulator contents and the contents of a local memory location. The instruction then places the result in the accumulator. The operand register specified by *d* provides the local memory address. The instruction clears the carry flag.

Instruction 032

IOP	AMPL	Description
032	A = A + (*dd*)	Adds contents of memory addressed by register *d* to A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits operand register data to the MA register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits the MA register data to the bank address registers. Send the read request signal to local memory. The function of CP 3 repeats until the acceptance signal is received.
CP 4	The IOP computation section receives the acceptance signal from local memory.
CP 5-6	The IOP computation section waits for data from local memory.
CP 7	Transmits local memory data to the addend register.
CP 8	Transmits data from the accumulator and the addend register to the adder and performs addition.
CP 9	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 032 adds the contents of a local memory location to the contents of the accumulator. The operand register specified by *d* provides the local memory address. The instruction complements the carry flag if the addition process propagates a carry flag.

Instruction 033

IOP	AMPL	Description
033	A = A - (*dd*)	Subtract contents of memory addressed by register *d* from A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits data from operand register *d* to the MA register. The function of CP 2 is delayed if the DP register full and DP = RP.
CP 3	Transmits MA register data to the bank address registers. Sends a Read Request signal to local memory. The function of CP 3 repeats until the acceptance signal is received.
CP 4	The IOP computation section receives the acceptance from local memory.
CP 5-6	The IOP computation section waits for data from local memory.
CP 7	Transmits local memory data to the addend register.
CP 8	Transmits data from the accumulator and the addend register to the adder; performs subtraction.
CP 9	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 033 subtracts the contents of a local memory location from the contents of the accumulator. The operand register specified by *d* provides the local memory address. The instruction performs subtraction by complementing the subtrahend and adding the result to the accumulator

contents in one's complement mode. A 1 is then added to the result. The instruction complements the carry flag if either addition process propagates a carry bit.

Instruction 034

IOP	AMPL	Description
034	(*dd*) = A	Transmits A to memory addressed by register *d*

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits data from operand register *d* to the MA register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits MA register data to the bank address registers. Transmit a copy of the accumulator data to the bank operand registers. Send the Write Request signal to memory. The function of CP 3 repeats until the IOP computation section receives an acceptance signal.
CP 4	The IOP computation section receives the acceptance signal from local memory. The accumulator is free for subsequent instructions after receiving the acceptance signal.
CP 5	The IOP computation section waits for local memory to write the data.
CP 6	The write operation is complete.

Description

Instruction 034 replaces the contents of a local memory location with the current contents of the accumulator. Operand register *d* provides the local memory address.

Instruction 035

IOP	AMPL	Description
035	(*dd*) = A + (*dd*)	Add memory addressed by register *d* to A, result to same memory location

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits data from operand register *d* to the MA register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits MA register data to the bank address registers. The data remains in the MA register. Sends the Read Request signal to local memory. The function of CP 3 repeats until an acceptance signal is received.
CP 4	The IOP computation section receives an acceptance signal from local memory. The MA register holds data from CP 4.
CP 5-6	The IOP computation section waits for data from local memory.
CP 7	Transmits local memory data to the addend register.
CP 8	Transmits data to the adder and perform addition.
CP 9	Transmits the adder result to the accumulator.
CP 10	Transmits MA register data to the bank address registers. Transmits a copy of the accumulator data to the bank operand registers. Sends a Write Request signal to memory. The function of CP 10 repeats until an acceptance signal is received.

CP 11 The IOP computation section receives an acceptance signal from local memory. The accumulator is free for subsequent instructions after receiving the acceptance signal.

CP 12 The IOP computation section waits for local memory to write the data.

CP 13 The write operation to local memory is complete.

Description

Instruction 035 replaces the contents of a local memory location with its previous contents plus the current accumulator contents. Operand register *d* provides the local memory address. The instruction performs addition using the accumulator in a 16-bit, two's-complement mode. The instruction complements the carry flag if the addition process propagates a carry bit. The result goes into local memory and also remains in the accumulator.

Instruction 036

IOP	AMPL	Description
036	(*dd*) = (*dd*) + 1	Transmits memory addressed by register *d*, adds 1, result to same memory location

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits data from operand register *d* to the MA register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits MA register data to the bank address registers. The data remains in the MA register. Sends a Read Request signal to memory. The function of CP 3 repeats until an acceptance signal is received. The instruction clears the carry flag and enters +1 in the accumulator.
CP 4	The IOP computation section receives the acceptance signal from local memory.
CP 5-6	The IOP computation section waits for data from local memory.
CP 7	Transmits local memory data to the addend register.
CP 8	Transmits data to the adder; performs addition.
CP 9	Transmits the adder result to the accumulator.
CP 10	Transmits MA register data to the bank address registers. Transmit a copy of the accumulator data to the bank operand registers. Sends a Write Request signal to local memory. The function of CP 10 repeats until the acceptance signal is received.

CP 11 The IOP computation section receives the acceptance signal from local memory. The accumulator is free for subsequent instructions after receiving the acceptance signal.

CP 12 The IOP computation section waits for local memory to write the data.

CP 13 The write operation to local memory is complete.

Description

Instruction 036 increments the contents of a local memory location by 1. Operand register *d* provides the local memory address. Initially, the instruction clears the accumulator and the carry flag. A +1 is entered in the accumulator. The content of the local memory location goes into the addend register and is then added to the accumulator contents. The result transfers to local memory and also remains in the accumulator. The instruction also complements the carry flag if the addition process propagates a carry bit.

Instruction 037

IOP	AMPL	Description
037	(*dd*) = (*dd*) - 1	Transmit memory addressed by register *d* to A, result to same memory location

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register data to the MA register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits the MA register data to the bank address registers. The data remains in the MA register. Sends a Read Request signal to local memory. The function of CP 3 repeats until an acceptance signal is received. The operation clears the carry flag and enters -1 in accumulator.
CP 4	The IOP computation section receives the acceptance signal from local memory.
CP 5-6	The IOP computation section waits for data from local memory.
CP 7	Transmits local memory data to the addend register.
CP 8	Transmits data to the adder; performs addition.
CP 9	Transmits the adder result to the accumulator.
CP 10	Transmits the MA register data to the bank address registers. Send the Write Request signal to local memory. Transmit a copy of the accumulator data to the bank operand registers. The function of CP 10 repeats until the acceptance signal is received.

CP 11 The IOP computation section receives the acceptance signal from local memory. The accumulator is free for subsequent instructions after receiving the acceptance signal.

CP 12 The IOP computation section waits for local memory to write the data.

CP 13 The write operation to local memory is complete.

Description

Instruction 037 decrements the contents of a local memory location by 1. The operand register specified by *d* provides the local memory address. Initially, the operation clears the carry flag and sets all accumulator bits. The contents of the local memory location are entered in the addend register and then added to the accumulator contents. The result transfers to the local memory location and also remains in the accumulator. The operation complements the carry flag if the addition process propagated a carry bit.

Instruction 040

IOP	AMPL	Description
040	C = 1, iod = DN	Sets carry equal 1 if channel *d* done

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

Instruction Timing

Cycle	Action
CP 0	Issue the instruction.
CP 1	Transmit the *d* field to the I/O channels.
CP 2-4	The IOP computation section waits for a response from the I/O channels.
CP 5	Force the state of the carry flag. The accumulator is free for subsequent instructions.

Description

Instruction 040 forces the carry flag to the same state as the done flag of the channel specified by *d*.

Insert a delay of 1 CP if issuing an I/O instruction that alters the busy or done flags before instruction 040.

The channel 000 done flag is always set. Set the carry flag by setting *d* = 000 in this instruction.

Instruction 041

IOP	AMPL	Description
041	C = 1, *iod* = BZ	Sets carry equal to 1 if channel *d* busy

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

Instruction Timing

Cycle	Action
CP 0	Issue the instruction.
CP 1	Transmit the *d* field to the I/O channels.
CP 2-4	The IOP computation section waits for a response from the I/O channels.
CP 5	Force the carry flag. The accumulator is free for subsequent instructions.

Description

Instruction 041 forces the carry flag to the same value as the channel *d* busy flag.

Insert a delay of 1 CP if issuing an I/O instruction that alters the busy or done flags before instruction 041.

The channel 000 busy flag is always clear. Clear the carry flag by issuing an 041 instruction with a *d* field of 000.

Instruction 042

IOP	AMPL	Description
042	C = 1, IOB = DN	Sets carry equal to 1 if channel B done

Issue Conditions

The accumulator is free 2 CPs after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B designator to the I/O channels.
CP 2-4	The IOP computation section waits for a response from the I/O channels.
CP 5	Forces the state of the carry flag. The accumulator is free for subsequent instructions.

Description

Instruction 042 forces the carry flag to the same value as the done flag of the channel specified by the contents of the B register.

Insert a delay of 1 CP if issuing an I/O instruction that alters the busy or done flags before instruction 042.

The channel 000 done flag is always set. Set the carry flag by setting B to 000 in this instruction.

Instruction 043

IOP	AMPL	Description
043	C = 1, IOB = BZ	Sets carry equal to 1 if channel B busy

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B designator to the I/O channels.
CP 2-4	The IOP computation section waits for a response from the I/O channels.
CP 5	Forces the carry flag. The accumulator is free for subsequent instructions.

Description

Instruction 043 forces the carry flag to the same value as the busy flag of the channel specified by the contents of the B register.

Insert a delay of 1 CP if issuing an I/O instruction that alters the busy or done flags before instruction 043.

The channel 000 busy flag is never set. Clear the carry flag by setting B to 000 in this instruction.

Instruction 044

IOP	AMPL	Description
044	A = A > B	Right shift C and A by B places, end off

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents to the addend register.
CP 2	Transmits data and the inverted B count to the shifter; performs the shift.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 044 shifts the contents of the accumulator and the carry flag to the right by B bit positions, treating the carry flag as the high-order bit of the accumulator contents. Zero values enter in the carry flag and propagate to the right as the shift progresses. No shift occurs if the shift count is 0. The operation clears the accumulator and carry flag if the shift count is greater than 16_{10}.

The low-order 5 bits of the addend register provide the shift count. The other bits are ignored.

Instruction 045

IOP	AMPL	Description
045	A = A < B	Left shifts C and A by B places, end off

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents to the addend register.
CP 2	Transmits data and the B-register contents to the shifter; performs the shift.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 045 shifts the contents of the accumulator and the associated carry flag to the left by B bit positions treating the carry flag as the 17th bit to the left of the accumulator contents. Zero values enter the low-order bit positions of the accumulator and propagate to the left as the shift progresses. Bits shifted from the carry flag are discarded. No shift occurs if the shift count is 0. The operation clears the accumulator and carry flag if the shift count is greater than 16_{10}.

The low-order 5 bits of the addend register provide the shift count. The other bits are ignored.

Instruction 046

IOP	AMPL	Description
046	A = A >> B	Right shifts C and A by B places, end off

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B register contents to the addend register.
CP 2	Transmits data and the inverted B register contents to the shifter; performs the shift.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 046 shifts the contents of the accumulator and the associated carry flag to the right in a circular mode by B bit positions, treating the carry flag as the 17th bit to the left of the accumulator contents. This shift does not discard any bits. Bits shifted from the right end of the accumulator are returned to the carry flag. No shift occurs if the shift count is 0.

The low-order 5 bits of the addend register provide the shift count. The other bits are ignored.

Instruction 047

IOP	AMPL	Description
047	A = A << B	Left shifts C and A by B places, circular

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents to the addend register.
CP 2	Transmits data and the B-register contents to the shifter; performs the shift.
CP 3	Transmits the shifter result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 047 shifts the contents of the accumulator and the carry flag to the left in a circular mode by B-bit positions. The operation treats the carry flag as the high-order bit to the left of the accumulator contents. This shift does not discard any bits. Bits shifted from the carry flag enter the right end of the accumulator. No shift occurs if the shift count is 0.

The low-order 5 bits of the addend register provide the shift count. The other bits are ignored.

Instruction 050

IOP	AMPL	Description
050	A = B	Transmits B to A

Hold Issue Conditions

The accumulator is free.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 050 enters the B-register contents in the accumulator as a 9-bit positive integer and clears the carry flag. The high-order accumulator bits are 0.

Instruction 051

IOP	AMPL	Description
051	A = A & B	Forms the logical product of A and B, result to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

The DP register is not full.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents and the accumulator contents to accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 051 forms the bit-by-bit logical product of the previous accumulator contents and the B-register contents, treating the B-register contents as a 9-bit positive integer. Circuitry at the input of the accumulator performs the logical product operation. The results go into the accumulator. The instruction clears the carry flag.

Instruction 052

IOP	AMPL	Description
052	A = A + B	Adds B to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents to the addend register.
CP 2	Transmits data to the adder; performs addition.
CP 3	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 052 adds the B-register contents to the previous accumulator contents in the 16-bit, two's-complement mode treating the B-register contents as a 9-bit positive integer. The instruction complements the carry flag if the addition process propagates a carry bit.

Instruction 053

IOP	AMPL	Description
053	A = A - B	Subtract B from A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issue the instruction.
CP 1	Transmit the B-register contents to the addend register.
CP 2	Transmit data to the adder; performs subtraction.
CP 3	Transmit the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 053 subtracts the B-register contents from the previous accumulator contents in 16-bit, two's-complement mode, treating the B-register contents as a 9-bit positive integer. The operation performs subtraction by complementing the contents of the addend register, adding the result to the previous accumulator contents, and adding a 1 to these results. The instruction complements the carry flag if either addition process propagates a carry bit.

Instruction 054

IOP	AMPL	Description
054	B = A	Transmits B to A

Hold Issue Conditions

The accumulator is free.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Enters accumulator data in the B-register.

Description

Instruction 054 replaces the B-register contents with the 9 low-order accumulator bits.

Instruction 055

IOP	AMPL	Description
055	B = A + B	Adds B to A, result to B

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents to the addend register. Reserves the B register.
CP 2	Transmits data to the adder; performs addition.
CP 3	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.
CP 4	Transmits a copy of the accumulator data to the B register. The B-register reservation is cleared.

Description

Instruction 055 adds the contents of the B register to the previous accumulator contents. The operation treats the B register contents as a 9-bit positive integer. Addition is performed in a 16-bit, two's-complement mode. The instruction complements the carry flag if a carry is propagated from the accumulator in the addition process. The results go into the B register at the end of the instruction.

Instruction 056

IOP	AMPL	Description
056	B = B + 1	Transmits B to A, adds 1, result to B

Hold Issue Conditions

The accumulator is free.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the contents of the B register to the addend register, enters a +1 in the accumulator, and clears the carry flag. The B register is reserved.
CP 2	Transmits data to the adder; performs addition.
CP 3	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.
CP 4	Transmits a copy of the accumulator data to the B register. The B-register reservation is cleared.

Description

Instruction 056 adds 1 to the contents of the B register and returns the results to both the B register and the accumulator. The carry flag is cleared at the beginning of this operation and a 1 is entered in the accumulator. The operation adds the B-register contents to the accumulator contents in 16-bit, two's-complement mode, treating the B register contents as a 9-bit positive integer. The low-order 9 bits of the accumulator are then returned to the B register.

Instruction 057

IOP	AMPL	Description
057	B = B - 1	Transmits B to A, subtracts 1, result to B

Hold Issue Conditions

The accumulator is free.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register contents to the addend register, enters a -1 in the accumulator, and clears the carry flag. The B register is reserved.
CP 2	Transmits the accumulator data to the adder; performs addition.
CP 3	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.
CP 4	Transmits a copy of the accumulator data to the B register. The B-register reservation is cleared.

Description

Instruction 057 subtracts 1 from the contents of the B register and returns the results to the B register and the accumulator. The carry flag is cleared at the beginning of this operation and all accumulator bits are set. The operation enters the contents of the B register in the addend register and then adds them to the accumulator contents, treating the B-register contents as a 9-bit positive integer. The operation complements the carry flag if the addition process propagates a carry bit. The low-order 9 bits of the accumulator are then returned to the B register.

Instruction 060

IOP	AMPL	Description
060	A = (B)	Transmits operand register B to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register data to the RP register.
CP 2	Transmits the operand register data to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 060 enters the contents of operand register B in the accumulator. The instruction then clears the carry flag.

Instruction 061

IOP	AMPL	Description
061	A = A & (B)	Logical product of A and operand register B to A

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register data to the RP register.
CP 2	Transmits the operand register data to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 061 forms the bit-by-bit logical product of the previous accumulator contents and the contents of operand register B, and places the result in the accumulator. It then clears the carry flag.

Circuitry at the input of the accumulator forms the logical product of the previous accumulator contents and the operand register. The logical product function does not require additional time.

Instruction 062

IOP	AMPL	Description
062	A = A + (B)	Adds operand register B to A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

The B register is not reserved.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register data to the RP register.
CP 2	Transmits the operand register data to the addend register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits data from the accumulator and addend register to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 062 adds the contents of operand register B to the previous accumulator contents in two's complement mode. The instruction complements the carry flag if the addition process propagates a carry bit.

Instruction 063

IOP	AMPL	Description
063	A = A - (B)	Subtracts operand register B from A

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

The B register is not reserved.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B register data to the RP register.
CP 2	Transmits the operand register data to the addend register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits data from the accumulator and addend register to the adder; performs subtraction.
CP 4	Transmits the adder result to the accumulator. The accumulator is free for subsequent instructions.

Description

Instruction 063 subtracts the contents of operand register B from the previous accumulator contents. Subtraction is performed by complementing the subtrahend, adding the result to the accumulator contents in two's complement mode, and adding a 1 to this result. The instruction complements the carry flag if either addition process propagates a carry bit.

Instruction 064

IOP	AMPL	Description
064	(B) = A	Transmits A to operand register B

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

The DP register is not full.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register data to the RP and DP registers.
CP 2	Transmits the accumulator data to the operand register. The accumulator is free for subsequent instructions.

Description

Instruction 064 stores the accumulator contents in operand register B.

Instruction 065

IOP	AMPL	Description
065	(B) = A + (B)	Adds operand register B to A, result to operand register B

Hold Issue Conditions

The accumulator is free 2 CPs after issue.

The DP register is not full.

The B register is not reserved.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register data to the RP and DP registers. Indicates that the DP register is full.
CP 2	Transmits the operand register data to the addend register. Discards the RP register pointer.
CP 3	Transmits data to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. Transmits the DP register data to the RP register, entering the same pointer discarded in CP 2. The accumulator is free for subsequent instructions. Indicates that the DP register is empty.
CP 5	Transmits a copy of the accumulator data to the operand register.

Description

Instruction 065 adds the contents of the accumulator to the contents of operand register B in two's complement mode. The instruction complements the carry flag if the addition process propagated a carry bit. The instruction then replaces the contents of operand register B with the new accumulator contents.

Instruction 066

IOP	AMPL	Description
066	(B) = (B) + 1	Transmits operand register to A, add 1, result to operand

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The DP register is not full.

The B register is not reserved.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register data to the RP and DP registers. Indicates that the DP register is full.
CP 2	Transmits the operand register data to the addend register. Enters a +1 in the accumulator. Discards the RP register pointer. Clears the carry flag.
CP 3	Transmits data to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. Transmits the DP register data to the RP register, entering the same pointer discarded in CP 2. Indicates that the DP register is empty. The accumulator is free for subsequent instructions.
CP 5	Transmits a copy of the accumulator data to the operand register.

Description

Instruction 066 adds 1 to the contents of operand register B and returns the result to the accumulator and the operand register. Initially, the carry flag is cleared and a 1 is entered in the accumulator. The operation transfers the operand register contents to the addend register and adds them to the accumulator contents in two's complement mode. The operation complements the carry flag if a carry is propagated from the

accumulator in the addition process. The result goes to the operand register.

Instruction 067

IOP	AMPL	Description
067	(B) = (B) - 1	Transmits operand register to A, subtract 1, result to operand register B

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The DP register is not full.

The B register is not reserved.

No DP-to-RP conflicts exist at issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the B-register data to the RP and DP registers. Indicates that the DP register is full.
CP 2	Transmits the operand register data to the addend register. Enters a -1 value in the accumulator. Discards the RP register pointer. Clears the carry flag.
CP 3	Transmits the data to the adder; performs addition.
CP 4	Transmits the adder result to the accumulator. Transmits the DP register data to the RP register, entering the same pointer discarded in CP 2. Indicates that the DP register is empty. The accumulator is free for subsequent instructions.
CP 5	Transmits a copy of the accumulator data to the operand register.

Description

Instruction 067 subtracts 1 from the contents of operand register B and returns the results to the accumulator and the operand register. Initially, the carry flag clears and all accumulator bits set. The operation enters the operand register contents in the addend register and adds them to the accumulator contents. The carry flag is complemented if the addition process propagates a carry bit. The result goes to the operand register.

Instruction 070

IOP	AMPL	Description
070	P = P + *d*	Jumps to P + *d*

Hold Issue Conditions

The next instruction parcel is available.

The adder is free (no delayed add operations are in progress).

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the contents of P to the background accumulator. Transmits the *d* field data to the background addend register.
CP 2	Transmits the background accumulator data to the adder; performs addition.
CP 3	Transmits the adder result to the P register. Transmits the first instruction in the new program sequence from the instruction stack to the CIP register if the branch is within the instruction stack. In this case, the first instruction in the new program sequence issues at CP 5.
CP 4	Generates a fetch request at the new P address if the branch is outside the instruction stack. A previous internal fetch request can delay this action *m* CPs.
CP *m*+1	The fetch operation generates a memory request. Memory conflicts can generate a delay of *n* CPs.
CP *n*+1	Memory sends an acceptance signal.
CP *n*+2	Transmits memory data to the CIP register.
CP *n*+3	The CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 070 terminates the current program sequence and begins a new sequence. The initial address for the new sequence equals the address of the current instruction plus the *d* field. The operation treats the *d* field as a 9-bit positive integer and performs addition in a 16-bit, two's-complement mode. The operation does not alter the accumulator contents or the carry flag.

Further instructions cannot issue until the initial instruction of the new program sequence issues.

Instruction 071

IOP	AMPL	Description
071	P = P - *d*	Jumps to P - *d*

Hold Issue Conditions

The next instruction parcel is available.

The adder is free (no delayed add operations are in progress).

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the contents of the P register to the background accumulator. Transmits the *d* field data to the background addend register.
CP 2	Transmits the background accumulator data to the adder: performs subtraction.
CP 3	Transmits the adder result to the P register. Transmits the first instruction in the new program sequence from the instruction stack to the CIP register if the branch is within the instruction stack. This instruction can issue at CP 5.
CP 4	Generates a fetch request at the new P register address if the branch is outside the instruction stack. A previous internal fetch request can delay this action *m* CPs.
CP *m*+1	The fetch operation generates a memory request. Memory conflicts can generate a delay of *n* CPs.
CP *n*+1	Memory sends an acceptance signal.
CP *n*+2	Transmits memory data to the CIP register.
CP *n*+3	The CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 071 terminates the current program sequence and begins a new sequence. The initial address for the new sequence equals the address of the current instruction minus the *d* field. The operation performs subtraction in a 16-bit, two's-complement mode, treating the *d* field as a 9-bit positive integer. The operation does not alter the accumulator contents or the carry flag.

Further instructions cannot issue until the initial instruction of the new program sequence issues.

Instruction 072

IOP	AMPL	Description
072	R = P + *d*	Jumps to P + *d*, returns on completion

Hold Issue Conditions

The next instruction parcel is available.

The adder is free (no delayed add operations are in progress).

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the contents of the P register to the background accumulator. Transmits the *d* field data to the background addend register.
CP 2	Transmits the background accumulator data to the adder; performs addition. Advances the E pointer by 1.
CP 3	Transmits the contents of P + 1 to the program exit stack. Transmits the adder result to the P register.
CP 4	Generates a fetch request at the new P address if the branch is outside the instruction stack. A previous internal fetch request can delay this action m CPs.
CP *m*+1	The fetch generates a memory request. Memory conflicts can generate a delay of n CPs.
CP *n*+1	Memory sends an acceptance signal.
CP *n*+2	Transmits memory data to the CIP register.
CP *n*+3	The CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 072 suspends execution of the current program sequence and calls a subprogram for execution by the following actions. The instruction advances the value of the E pointer by 1 and stores the address of the next sequential instruction of this program sequence in the program exit stack. Then, the operation begins executing a new program sequence. The initial address for the new sequence equals the address of the current instruction plus the *d* field. The operation treats the *d* field as a 9-bit positive integer and adds in a 16-bit, two's-complement mode. The operation does not alter the accumulator contents or the carry flag.

Instruction 073

IOP	AMPL	Description
073	R = P - *d*	Jumps to P - *d*, returns on completion

Hold Issue Conditions

The next instruction is available.

The adder is free (no delayed add operations in progress).

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the contents of the P register to the background accumulator. Transmits the *d* field data to the background addend register.
CP 2	Transmits the background accumulator data to the adder; perform subtraction. Advances the value of E by 1.
CP 3	Transmits the contents of P + 1 to the program exit stack. Transmits the adder result to the P register.
CP 4	Generates a fetch request at the new P address if the branch is outside the instruction stack. A previous internal fetch request can delay this action m CPs.
CP *m*+1	The fetch operation generates a memory request. Memory conflicts can generate a delay of *n* CPs.
CP *n*+1	Memory sends an acceptance signal.
CP *n*+2	Transmits memory data to the CIP register.
CP *n*+3	The CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 073 suspends execution of the current program sequence and calls a subprogram for execution. The instruction advances the value of E by 1 and stores the address of the next sequential instruction of this program sequence in the program exit stack. Then, the instruction begins a new program sequence. The initial address for the new sequence equals the address of the current instruction minus the *d* field. The operation treats the *d* field as a 9-bit positive integer and performs subtraction in a 16-bit, two's-complement mode. The instruction does not alter the accumulator contents or the carry flag.

Instruction 074

IOP	AMPL	Description
074	P = *dd*	Jumps to address in operand register *d*

Hold Issue Conditions

The next instruction parcel is available.

No DP-to-RP conflicts exist at issue.

No 150 through 153 or 170 through 173 instructions were issued in the last CP.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register contents to the background accumulator. Enters 0 in the background addend register. The function of CP 2 is delayed if the DP register is full and DP = RP.
CP 3	Transmits the background accumulator data to the adder; adds. The function of CP 3 is delayed if the adder is busy.
CP 4	Transmits the adder result to the P register. Generates a fetch request at the new P address. A previous internal fetch request can delay this action *m* CPs.
CP *m*+1	The fetch operation generates a memory request. Memory conflicts can generate a delay of n CPs.
CP *n*+1	Local memory sends an acceptance signal.
CP *n*+2	Transmits the memory data to the CIP register.
CP *n*+3	The CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 074 terminates the current program sequence and begins a new sequence. Operand register *d* provides the initial address for the new sequence. Further instructions cannot issue until the first instruction of the new program sequence is in the CIP register.

The instruction sets the program fetch request flag if the contents of operand register *d* equals 0.

Instruction 075

IOP	AMPL	Description
075	P = *dd* + *k*	Jumps to sum of *k* and operand register *d*

Hold Issue Conditions

The next instruction parcel is available.

No DP-to-RP conflicts exist at issue.

No 150 through 153 or 170 through 173 instructions issued in the last CP.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register contents to the background accumulator. Transmits the constant *k* to the background addend register. The operand register transfer is delayed if the DP register is full and DP = RP.
CP 3	Transmits the background accumulator contents to the adder; adds. The function of CP 3 is delayed if the adder is busy.
CP 4	Transmits the adder result to the P register. Generates a fetch request at the new P address. A previous internal fetch request can delay this action *m* CPs.
CP *m*+1	The fetch operation generates a memory request. Memory conflicts can generate a delay of *n* CPs.
CP *n*+1	Memory sends an acceptance signal.
CP *n*+2	Transmits memory data to the CIP register.
CP *n*+3	CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 075 terminates the current program sequence and begins a new sequence. The initial address for the new sequence equals the next parcel of the current program sequence plus the contents of operand register *d*. The operation performs addition in a 16-bit, two's-complement mode. The operation does not alter the contents of the accumulator or the carry flag. Further instructions cannot issue until the first instruction of the new program sequence is in the CIP register.

This instruction sets the program fetch request flag if the contents of operand register *d* contains 0.

Instruction 076

IOP	AMPL	Description
076	R = *dd*	Jumps to address in operand register *d*, returns on completion

Issue Conditions

The next instruction parcel is available.

No DP-to-RP conflicts exist at issue.

No 150 through 153 or 170 through 173 instruction issued in the last CP.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register *d* contents to the background accumulator. Enters a value of 0 in the background addend register. Operand register transfer is delayed if the DP register is full and DP = RP.
CP 3	Transmits the background accumulator data to the adder; adds. Advances the E pointer by 1. The function of CP 3 is delayed if the adder is busy.
CP 4	Transmits the contents of the address specified by the P register contents plus 1 to the program exit stack. Transmits the adder result to the P register. Generates a fetch request at the new P address. A previous internal fetch request can delay this action *m* CPs.
CP *m*+1	The fetch operation generates a memory request. Memory conflicts can delay this action *n* CPs.
CP *n*+1	Memory sends an acceptance signal.
CP *n*+2	Transmits memory data to the CIP register.
CP *n*+3	CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 076 suspends execution of the current program sequence and calls a subprogram for execution. The instruction advances the value of the E pointer by 1, stores the address of the next sequential instruction of this sequence in the program exit stack, and begins a new program sequence. Operand register *d* provides the initial address for the new sequence.

The program exit stack boundary flag sets if the advanced E value is 14. The program fetch request flag sets if the contents of operand register *d* is 0. Further instructions cannot issue until the first instruction of the new sequence is in the CIP register.

Instruction 077

IOP	AMPL	Description
077	R = *dd* + *k*	Jumps to address sum of *k* and operand register *d*, returns on completion

Hold Issue Conditions

The next instruction parcel is available.

No DP-to-RP conflicts exist at issue.

No 150 through 153 or 170 through 173 instructions issued in the last CP.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits register number *d* to the RP register.
CP 2	Transmits the operand register contents to the background accumulator. Transmits the constant *k* to the background addend register. Operand register transfer is delayed if the DP register is full and DP = RP.
CP 3	Transmits the background accumulator data to the adder; performs addition. Advances the E pointer by 1. The CP 3 function is delayed if the adder is busy.
CP 4	Transmits the contents of the address specified by the P register plus 2 to the program exit stack. Transmits the adder result to the P register. Generates a fetch request at the new P address. A previous internal fetch request can delay this action *m* CPs.
CP *m*+1	The fetch operation generates a memory request. Memory conflicts can generate a delay of *n* CPs.
CP *n*+1	Memory sends an acceptance signal.
CP *n*+2	Transmits memory data to the CIP register.
CP *n*+3	The CIP register data is available for decoding.
CP *n*+4	Issues the first instruction of the new sequence.

Description

Instruction 077 suspends the current program sequence and calls a subprogram for execution. The instruction advances the value of the E pointer by 1 and stores the address that is 2 greater than the address of the current instruction in the program exit stack. The instruction begins a new program sequence. The initial address for the new sequence equals the contents of operand register *d* plus the next parcel of the current program sequence. Addition is performed in a 16-bit, two's-complement mode. The operation does not alter the contents of the accumulator or the carry flag.

The program exit stack boundary flag sets if the advanced E value is 14. The program fetch request flag sets if the contents of operand register *d* is 0. Further instructions cannot issue until the initial instruction of the new sequence issues.

IOP	AMPL	Description
100	P = P + *d*, C = 0	Jumps to P + *d* if carry = 0
101	P = P + *d*, C # 0	Jumps to P + *d* if carry # 0
102	P = P + *d*, A = 0	Jumps to P + *d* if A = 0
103	P = P + *d*, A # 0	Jumps to P + *d* if A # 0
104	P = P - *d*, C = 0	Jumps to P - *d* if carry = 0
105	P = P - *d*, C # 0	Jumps to P - *d* if carry # 0
106	P = P - *d*, A = 0	Jumps to P - *d* if A = 0
107	P = P - *d*, A # 0	Jumps to P - *d* if A # 0
110	R = P + *d*, C = 0	Jumps to P + *d* if carry = 0, returns on completion
111	R = P + *d*, C # 0	Jumps to P + *d* if carry # 0, returns on completion
112	R = P + *d*, A = 0	Jumps to P + *d* if A = 0, returns on completion
113	R = P + *d*, A # 0	Jumps to P + *d* if A # 0, returns on completion
114	R = P - *d*, C = 0	Jumps to P - *d* if carry = 0, returns on completion
115	R = P - *d*, C # 0	Jumps to P - *d* if carry # 0, returns on completion
116	R = P - *d*, A = 0	Jumps to P - *d* if A = 0, returns on completion
117	R = P - *d*, A # 0	Jumps to P - *d* if A # 0, returns on completion
120	P = *dd*, C = 0	Jumps to address in operand register *d* if carry = 0
121	P = *dd*, C # 0	Jumps to address in operand register *d* if carry = 0
122	P = *dd*, A = 0	Jumps to address in operand register *d* if A = 0
123	P = *dd*, A # 0	Jumps to address in operand register *d* if A # 0
124	P = *dd* + *k*, C = 0	Jumps to address in operand register *d* + *k* if carry = 0
125	P = *dd* + *k*, C # 0	Jumps to address in operand register *d* + *k* if carry # 0
126	P = *dd* + *k*, A = 0	Jumps to address in operand register *d* + *k* if A = 0
127	P = *dd* + *k*, A # 0	Jumps to address in operand register *d* + *k* if A = 0
130	R = *dd*, C = 0	Jumps to address in operand register *d* if carry = 0, returns on completion
131	R = *dd*, C # 0	Jumps to address in operand register *d* if carry # 0, returns on completion
132	R = *dd*, A = 0	Jumps to address in operand register *d* if A = 0, returns on completion
133	R = *dd*, A # 0	Jumps to address in operand register *d* if A # 0, returns on completion
134	R = *dd* + *k*, C = 0	Jumps to address in operand register *d* + *k* if carry = 0, returns on completion
135	R = *dd* + *k*, C # 0	Jumps to address in operand register *d* + *k* if carry # 0, returns on completion

IOP	AMPL	Description
136	R = *dd* + *k*, A = 0	Jumps to address in operand register *d* + *k* if A = 0, returns on completion
137	R = *dd* + *k*, A # 0	Jumps to address in operand register *d* + *k* if A # 0

Hold Issue Conditions

A given instruction has the same issue conditions as the corresponding unconditional branch instruction (instructions 070 through 077).

Instruction Timing

The instructions have the same timing as the corresponding unconditional instruction, if the branch is taken. Formation of the branch condition requires 1 CP after the accumulator or the carry flag becomes free. The issue of the next instruction waits until the branch criterion is available. If the branch criterion is available in CP 0 and the branch is not taken based on that criterion, the next instruction in the current program sequence can issue in the next CP.

Description

Instructions 100 through 137 are branch instructions that jump to a new program sequence only if a branch condition is met. Instructions 070 through 077 represent the eight branch modes in the unconditional form. All possibilities of these eight modes are combined with four branch criteria to form the set of instructions 100 through 137. The branch criteria are as follows:

(Any branch instruction 070 through 077), C = 0
(Any branch instruction 070 through 077), C # 0
(Any branch instruction 070 through 077), A = 0
(Any branch instruction 070 through 077), A # 0

The C = 0 branch condition causes the branch to be taken if the carry flag equals 0. If the carry flag equals 1, the current program sequence continues.

The C # 0 branch condition causes the branch to be taken if the carry flag equals 1. If the carry flag equals 0, the current sequence continues.

The A = 0 branch condition causes the branch to be taken if the accumulator content equals 0. If the accumulator content does not equal 0, the current sequence continues.

The A # 0 branch condition causes the branch to be taken if the accumulator content does not equal 0. If the accumulator content does equal 0, the current sequence continues.

I/O Channel Instructions 140 through 177

Instructions 140 through 177 allow IOP control of the I/O channel activity. The *d* field in instructions 140 through 157 specifies the I/O channel. The contents of the B register specifies the channel for instructions 160 through 177.

An I/O instruction sends the low-order 4 bits of the function code and a Go-function signal to the channel interface control circuitry. The channel interface interprets the 4-bit code in a manner unique to that channel.

I/O instructions provide accumulator data to a specific interface and transfer data from the interface to the accumulator.

Instructions 150 through 153 and 170 through 173 transfer a 16-bit quantity into the accumulator. The value transferred depends on which channel interface is used.

I/O channel instructions do not significantly delay the execution of the program sequence. The I/O channel control has no mechanism to delay execution of further instructions as a result of interpreting the 4-bit code. Delays in program functions must be programmed by sampling the busy and done flags or using the interrupt mechanism.

Allow 1 CP after issue of an I/O instruction before checking the busy or done flags. Allow 1 CP after issue of a channel function 6 or 7 before checking the channel interrupt number (IOR : 10).

Instructions 140 through 147 and 154 through 157

IOP	AMPL	Description
140	*iod* : 0	Channel *d* function 0
141	*iod* : 1	Channel *d* function 1
142	*iod* : 2	Channel *d* function 2
143	*iod* : 3	Channel *d* function 3
144	*iod* : 4	Channel *d* function 4
145	*iod* : 5	Channel *d* function 5
146	*iod* : 6	Channel *d* function 6
147	*iod* : 7	Channel *d* function 7
154	*iod* : 14	Channel *d* function 14
155	*iod* : 15	Channel *d* function 15
156	*iod* : 16	Channel *d* function 16
157	*iod* : 17	Channel *d* function 17

Hold Issue Conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1-2	Transmits the accumulator contents and function code to I/O channel *d*.
CP 3	The interface receives the information.

Description

These instructions send data to the interface defined by the contents of the *d* field.

Instructions 150 through 153

IOP	AMPL	Description
150	*iod* : 10	Channel *d* function 10
151	*iod* : 11	Channel *d* function 11
152	*iod* : 12	Channel *d* function 12
153	*iod* : 13	Channel *d* function 13

Hold Issue conditions

The accumulator is free 1 CP after issue.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the accumulator contents and the function code to I/O channel *d*.
CP 2	Clears the carry flag.
CP 3	The interface receives the information.
CP 4-5	The IOP waits for a response from the interface.
CP 6	The interface data goes to the accumulator. The accumulator is free for subsequent instructions.

Description

These instructions send data to the interface defined by the *d* field. The interface responds by returning a value to the accumulator. The carry flag clears.

Instructions 160 through 167 and 174 through 177

IOP	AMPL	Description
160	IOB : 0	Channel B function 0
161	IOB : 1	Channel B function 1
162	IOB : 2	Channel B function 2
163	IOB : 3	Channel B function 3
164	IOB : 4	Channel B function 4
165	IOB : 5	Channel B function 5
166	IOB : 6	Channel B function 6
167	IOB : 7	Channel B function 7
174	IOB : 14	Channel B function 14
175	IOB : 15	Channel B function 15
176	IOB : 16	Channel B function 16
177	IOB : 17	Channel B function 17

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1-2	Transmits the accumulator contents and the function code to I/O channel B.
CP 3	The interface receives the information.

Description

These instructions send data to the interface defined by the channel number in the B register.

Instructions 170 through 173

IOP	AMPL	Description
170	IOB : 10	Channel B function 10
171	IOB : 11	Channel B function 11
172	IOB : 12	Channel B function 12
173	IOB : 13	Channel B function 13

Hold Issue Conditions

The accumulator is free 1 CP after issue.

The B register is not reserved.

Instruction Timing

Cycle	Action
CP 0	Issues the instruction.
CP 1	Transmits the accumulator contents and the function code to I/O channel B.
CP 2	Clears the carry flag.
CP 3	The interface receives the information.
CP 4-5	The IOP waits for a response from the interface.
CP 6	The interface data goes to the accumulator. The accumulator is free for subsequent instructions.

Description

These instructions send data to the interface defined by the channel number in the B register. The interface responds by returning a value to the accumulator. The carry flag clears.

8 - IOS DIAGNOSTIC MODES

This section describes the following diagnostic capabilities for the I/O subsystem (IOS):

- The jump history log
- Instruction stack parity testing
- Local memory single-error correction/double-error detection (SECDED) testing
- Program exit stack parity testing

IOS diagnostic modes provide the jump history log, instruction stack parity testing, and local memory SECDED testing. Instructions 140 through 177 reference the diagnostic modes using channel designators 400 through 417. Each diagnostic mode corresponds to a channel designator. Table 8-1 lists and describes the diagnostic modes. A mode remains in effect until the next instruction sets the next mode.

Ensure that the SYSTEM/MAINT. switch on the IOS control panel is in the MAINT. position before setting the modes in group B. The modes in group A can be issued at any time.

JUMP HISTORY LOG

The jump history log records the addresses of the last 4,096 branches taken by an IOP. The log records the address of the parcel that follows the branch.

Modes 406 and 407 turn the log on and off. Allow about 4 clock periods (CPs) to turn the log on or off before the next branch is logged.

Diagnostic modes 404 and 405 have two functions depending on whether the log is on or off. After mode 406 turns the log on, mode 404 causes all branches to be logged, and channel 405 causes only return jumps to be logged. After mode 407 turns the history log off, mode 404 selects incrementing read operation and mode 405 selects decrementing read operation. The IOP always writes data into the log in incrementing addresses, but reads out data either in incrementing addresses (first in, first out) or decrementing addresses (last in, first out).

The log is a cyclic buffer. Therefore, the only method of determining the address currently being read or written is to count the number of branches since the last clear history log address mode.

Table 8-1. Diagnostic Modes

Mode Designator	Cleared by Mode	Default	Mode Description
			Set A
400	--		Not used
401	--		Not used
402	--		Not used
403	404-407		Clears history log address
404	405		Logs all jumps (history log on) Incrementing read operation (history log off)
405	404	On	Logs return jumps (history log on) Decrementing read operation (history log off)
406	407		Turns history log off
407	406	On	Turns history log on
			Set B
410	410-417		Instruction stack parity mode
411	410-417		Clears error flags
412	410-417		Disables SECDED writing of check bits and error correction
413	410-417		Reads SECDED information into the B register (syndrome bits and error location)
414	--		Not used
415	--		Not used
416	410-417		Reads SECDED information into the B register (uncorrectable error bit and error-byte location)
417	410-417	On	Disables modes 410 through 416

The log has adequate buffering to log two consecutive branches. If three consecutive branches occur, the second branch address is not logged.

Use the PSX : 13 function (instruction 153002) to read the jump history log. The log uses the first 153002 instruction in an incrementing read and the first two 153002 instructions of a decrementing read for syncronization. Disregard the results of these instructions. Do not allow either return jumps or interrupts at any time during the log read-out sequence because the readout uses the program exit stack. To re-enable normal program exit stack operation, issue a PSX : 0 function (instruction 140002).

NOTE: Observe the following timing requirements:

- Allow 4 CPs between consecutive read or write operations to the jump history log. Out-of-stack jumps require more than 4 CPs, but in-stack relative branches require only 3 CPs.

- Allow 4 CPs between a turn history log off operation (mode 406) and a decrementing read operation (mode 405).

- Allow 4 CPs between a decrementing read operation (mode 405) and a PSX : 13 function (instruction 153002).

- Allow 8 CPs between consecutive PSX : 13 functions (instruction 153002).

INSTRUCTION STACK PARITY TESTING

Diagnostic modes 410 and 411 test the instruction stack parity capability. While mode 410 is in effect, B-register bit 2^8 selects the parity bit (1 = odd parity, 0 = even parity), and B register bits 2^0 and 2^1 equal the even and odd parcel error flags, respectively. Mode 411 clears the even and odd parcel error flags.

Mode 410 does not go into effect immediately when the instruction that selects mode 410 issues. The next out-of-stack branch enables the mode. Mode 410 remains in effect until the next B register instruction is issued (42 through 67, 160 through 177). Up to 12 parcels can be fetched and written before the B register instruction. The B register instruction deactivates mode 410. However, a subsequent out-of-stack branch re-enables the mode if it occurs before a mode 410 through 417 instruction.

Clearing the B register does not clear accumulator bits 2^0 and 2^1 if the error latches are set. Diagnostic mode 411 clears the error latches.

Perform the following sequence to test instruction stack parity:

1. Format the instructions to be used in the test into the buffer. Allow 16 parcels after the B register instruction for padding of the fetch-ahead mechanism. For testing and padding, use the 000 and 050 instructions because the lower 9 bits are not used. An 014 instruction allows testing of all 16 bits in a parcel.

2. Branch out of stack to the buffer to enable write parity.

3. Issue the test instructions formatted in Step 1.

4. Issue a B register instruction to stop write parity from using B register bit 8.

5. Issue pad instructions (some of which are written with B register bit 8 as their parity).

6. Clear the B register if an instruction was issued to write to the B register.

7. Read the error flags out of B register bits 2^0 and 2^1.

8. Clear the error flags by issuing a mode 411 instruction.

9. Issue a branch instruction.

LOCAL MEMORY SECDED TESTING

Diagnostic mode channels 412, 413, and 416 test local memory SECDED operation. Mode channel 412 disables the writing of check bits. This allows testing of the SECDED hardware by causing SECDED errors on the data subsequently read out of local memory.

Use the following sequence to cause local memory SECDED errors:

1. Write data into local memory; SECDED circuitry stores check bits with the data.

2. Issue a mode 412 instruction to disable writing of check bits.

3. Write data into local memory; since Step 2 disabled writing of check bits, this data is stored with the check bits of Step 1.

4. Enable error correction and logging into the B register using mode 413.

5. Read data out of local memory; reading out the data with the erroneous check bits causes a SECDED error. Diagnostic mode 413 causes the error to be logged in the B register. Table 8-2 shows the B-register bit assignments following a mode 413 instruction.

SECDED circuitry corrects single-bit errors and detects double-bit errors when it is functioning correctly. Errors are logged in the B register. B register bits 2^0 through 2^4 contain the SECDED syndrome bits, and bits 2^5 through 2^8 contain the location of the parcel in error. Table 8-2 shows the bit assignments.

Table 8-2. B Register Contents for Diagnostic Mode 413

Bit	Description
2^0	Syndrome bit 0
2^1	Syndrome bit 1
2^2	Syndrome bit 2
2^3	Syndrome bit 3
2^4	Syndrome bit 4
2^5	Bank bit 0
2^6	Bank bit 1
2^7	Section bit 0
2^8	Section bit 1

Mode 416 enters the remaining error information into the B register. Table 8-3 shows the bit assignments. Always clear the B register between mode 413 and mode 416 instructions.

Table 8-3. B Register Contents for Diagnostic Mode 416

Bit	Description
2^0	Indicates the byte in error: 0 = lower byte 1 = upper byte
2^1	Non-correctable error
2^2	Always 0
2^3-2^8	Not changed

Diagnostic mode operation replaces the contents of the B register every time an error occurs. Therefore, the mode 416 errors match the mode 413 errors only if no errors occur between the two B register read operations.

A minimum of 16 CPs elapse between the error and the time the information becomes available in the B register. A single-bit error is available in the B register a maximum of 20 CPs after the error.

Use mode 413 and the B register used to identify an error when running a local memory diagnostic program. While operating in mode 413, the B register remains empty unless a SECDED error occurs. Clear the B register before running the diagnostic. If the B register contains anything at the end of the diagnostic, an error occurred.

Use the following sequence to ensure that mode 413 error information matches the mode 416 error information:

1. Set mode 417.
2. Read the B register (get 9 bits of information).
3. Clear B register.
4. Set mode 416.
5. Read the B register (get 2 bits of information).
6. Set mode 417 or 413 and clear the B register.

Syndrome codes are as follows:

Syndrome Code	Bit in Error
23	0
31	1
32	2
33	3
34	4
15	5
26	6
07	7

PROGRAM EXIT STACK PARITY TESTING

Channel function PSX : 16 provides testing of the program exit stack parity circuitry. Parity circuitry maintains 4 parity bits for each stack location. A single parity bit corresponds with 4 data bits as shown below:

Data Bits	Parity Bit
2^0-2^3	0
2^4-2^7	1
2^8-2^{11}	2
2^{11}-2^{15}	3

Enable interrupts when testing the exit stack parity circuits. Do not perform return jumps while the stack is being tested.

One way to write the test is as a series of nested loops. Test patterns should include data that causes errors and data that does not cause errors.

Force parity bits to 1's or 0's to force a parity error at a particular location on the stack. Verify that when the location is read, it does cause a stack error. If the parity circuitry operates correctly, the data written into the stack at any location only forces 1 parity bit at a time to cause an error. This proves that the associated checking circuitry is the circuitry reporting the error.

Turn off the diagnostic mode and rewrite all tested locations to remove any parity errors that may remain in the exit stack. Write random data to the stack and read it back to verify that no data-sensitive patterns remain.

GLOSSARY

GLOSSARY

A

Accumulator — A 16-bit register that stores operands or results.

Accumulator channel — An I/O channel that transfers data into and out of the accumulator but does not have direct access to local memory.

Addend register — A 16-bit register that supplies operands to the adder and shifter functional units.

Adder functional unit — A hardware section of an IOP that performs addition and subtraction.

B

B register — A 9-bit register that contains a pointer to an operand register, an operand, or an I/O channel address.

BMC-4, BMC-5 — Block multiplexer controller. The BMC-4 or BMC-5 provides a hardware interface to IBM and IBM-compatible peripheral controllers and their attached peripheral devices.

BIOP — Buffer I/O processor. One of the IOPs in the IOS. It performs disk input and output to and from Cray disk storage units attached to its channels.

Buffer memory — A random access solid-state memory shared by all IOPs in an IOS chassis.

Busy flag — A 1-bit register controlled by the channel interface functions. The flag generally sets when function execution begins and clears when the function is complete.

Bypass mode — A mode of operation in the IOS buffer memory and Cray mainframe central memory channels that allows direct transfer between buffer memory and central memory.

C

Carry bit register	A 1-bit register that holds the carry bit generated in the adder or shifter functional units.
Central memory	Memory in the Cray mainframe that is shared by all central processor units (CPUs) in the mainframe.
Channel interrupt enable flag	Enables or disables interrupts on a specific channel.
Check bits	Five bits generated by SECDED for each byte written into local memory and buffer memory.
Circular shift	A shift operation that causes bits shifted out of the shifter to be returned to the opposite side.
CP	Clock period. A cycle time of 12.5 ns used by the IOS.
CIP register	Current instruction parcel register. A register in the IOP control section that holds a decoded instruction until all issue conditions have been met and generates all control signals for the instruction.

D

d	The designator field of an instruction.
Dead dump	A listing from memory taken after a crash or shutdown.
Deadstart	The sequence of operations required to start an operating system program.
DMA	Direct memory access ports and channels. Local memory ports and corresponding I/O channels that transfer data directly into and out of memory.
DIOP	Disk I/O processor. One of the IOPs in the IOS. It performs disk input and output to and from Cray disk storage units attached to its channels.
DSU	Disk storage unit. These units store data on magnetic disks and are controlled by channel functions from the IOP.

Done flag — A 1-bit register controlled by the channel interface functions. The flag generally clears when function execution begins and sets when the function is complete.

DP register — Destination pointer register. This register is the source of the operand register pointer for the second of two successive instructions that use the same operand register.

E

E pointer — A 4-bit register that addresses locations in the program exit stack.

End-off shift — A shift operation that causes bits shifted out of the shifter to be dropped.

F

f — The function field of an instruction.

I

Instruction stack — A 32-bit circular buffer that contains instructions from local memory that are not yet executed.

IOP — I/O processor. A 16-bit processor with its own local memory that operates independantly in the IOS.

IOS — I/O subsystem. Provides high-capacity data communications between the central memory of a Cray mainframe and peripheral devices such as data storage devices, and front-end computers. The IOS includes two to four IOPs and buffer memory.

K

k — The constant field of an instruction.

L

Local memory — A random-access, solid-state memory. Each IOP has its own local memory.

M

Master clear	An operation that raises the master clear signal to the IOP and interrupts the program being executed.
MA register	Memory address register. A 16-bit register that holds the address for a read or write local memory reference.
MIOP	Master I/O processor. One of the IOPs in the IOS. It initializes the contents of buffer memory and deadstarts the other processors.
Monitor program	A common block of instructions that outputs memory contents to the maintenance control console (MCC), displays interrupt information, and builds memory error tables.

N

NIP register	Next instruction parcel register. A 16-bit register that holds the instruction before it enters the current instruction parcel register. Instruction decoding begins in this register.

O

Operand registers	A series of 512 registers that act as storage locations for data, as index registers, and as indirect memory address registers.

P

P register	Program address register. A 16-bit register that holds the local memory address of the instruction currently waiting to issue in the CIP register.
Peripheral expander	Interfaces peripherals such as disk drives, tape units, and printer/plotters to one DMA port of the MIOP.
Program exit stack	A set of 16 last-in, first-out registers that stores return addresses for program subroutine calls and interrupt routines.
Program exit stack flag	Indicates that the program exit stack is empty or full, that a parity error occurred in the stack, or that the program attempted to read an empty stack location.

Program fetch request flag	Provides a mechanism for calling the IOP monitor program when a new section of program code is required.
R	
RP register	Reference pointer register. This register directly addresses an operand register for reading or writing.
RTC	Real-time clock. This is a 17-bit counter/timer that interrupts the IOP at 1-ms intervals, and increments every CP.
S	
SECDED	Single-error correction/double-error detection. A method of error detection and correction used in local memory and buffer memory.
Shifter functional unit	A hardware section of an IOP that performs performs shifts to the left or right of up to 31 places.
SSD solid-state storage device	A device used for temporary data storage.
System interrupt enable flag	Enables or disables interrupts for the entire system.
X	
XIOP	Auxiliary I/O processor. One of the IOPs in an IOS. Its primary function is to receive and transmit data to and from IBM-compatible tape drives through block multiplexer controllers.

Reader Comment Form

Title: I/O Subsystem Models C and D System Programmer Reference Manual

Number: CSM-1009-000

Your comments help us improve the quality and usefulness of your publications. Please us the space provided below to share with us your comments. When possible, please give specific page and paragraph references.

NAME_____

JOB TITLE_____

FIRM_____

ADDRESS_____

CITY_____STATE_____ZIP_____

DATE_____

CRAY RESEARCH, INC.

CUT ALONG THIS LINE

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY CARD

FIRST CLASS PERMIT NO 6184 ST. PAUL, MN

POSTAGE WILL BE PAID BY ADDRESSEE

CRAY RESEARCH, INC.

Attention: HARDWARE PUBLICATIONS
770 Industrial Boulevard
Chippewa Falls, WI 54729

STAPLE

Cray Research, Inc.
Hardware Publications
770 Industrial Boulevard
Chippewa Falls, WI 54729