Jacquard Systems

J500 Videocomputer System

Reference Manual

JPS **500-01**

JPS500-01

1 APRIL 1979

REFERENCE MANUAL

J500 VIDEOCOMPUTER SYSTEM

Changed 15 October 1979

REVISION RECORD

REVISION	DESCRIPTION
	Manual Released - 1 April 1979
1	Revised and Reprinted 15 October 1979
2	Revised Page 1-2 and Figure 1-2 - 15 June 1980

A Comment Sheet is included at the back of this manual.

TABLE OF CONTENTS

SECTION 1. INTRODUCTION 1-1

1-1 Introduction. 1-1
1-2 Physical/Functional Organization 1-2
1-3 Physical Organization. 1-2
1-4 Functional Organization. 1-4
1-5 System Features 1-4
1-6 Expansion Capabilities 1-4
1-7 User Interface 1-4
1-8 Hardware Features. 1-4
1-9 Diskettes 1-4
1-10 Peripheral and Communication Options . 1-4

SECTION 2. INSTRUCTION SET 2-1

2-1 Introduction. 2-1
2-2 Arithmetic and Logic Units Referenced in Instructions. 2-1
2-3 Push-Down Stack Register (LIFO) 2-1
2-4 Status Register 2-1
2-5 Program Counter (PC) 2-1
2-6 Accumulators 0, 1, 2, and 3 (AC0, AC1, AC2, and AC3) . . . 2-2
2-7 Data and Instructions 2-2
2-8 Data Representation 2-2
2-9 Instructions. 2-2
2-10 Memory Addressing 2-2
2-11 Base Page Addressing. 2-2
2-12 Program-Counter Relative Addressing 2-2
2-13 Indexed Addressing. 2-2
2-14 Indirect Addressing 2-2
2-15 Notations and Symbols Used In Microprocessor Instruction Descriptions . . . 2-2
2-16 Instruction Descriptions 2-4
2-17 Load and Store Instructions 2-4
2-18 Arithmetic Instructions. 2-4
2-19 Logical Instructions 2-5
2-20 Skip Instructions 2-6
2-21 Transfer-of-Control Instructions 2-7
2-22 Shift Instructions 2-10
2-23 Register Instructions 2-13
2-24 Input/Output Instructions. 2-15
2-25 Halt and Flag Instructions 2-18
2-26 Interrupt System 2-20
2-27 Device Controllers. 2-20
2-28 Machine Stack Overflow. 2-20

SECTION 3. CENTRAL PROCESSOR 3-1

3-1 Introduction. 3-1
3-2 Microprogram Memory. 3-1
3-3 Microprogram Control Logic 3-1
3-4 Main Memory 3-1
3-5 Register File. 3-1
3-6 Arithmetic/Logic Unit (ALU) 3-1
3-7 Input Multiplexor and Condition-Bit. . . . 3-2
3-8 Output Flip-Flops and Flag RAM. 3-2
3-9 Peripheral Buffer Registers. 3-2
3-10 Programmable Sync/Async Communication Chips 3-2
3-11 CRT Character Generator Memory 3-2
3-12 J500 Microprogram Description 3-2

SECTION 4. CRT AND KEYBOARD 4-1

4-1 Keyboard 4-1
4-2 CRT Display 4-1
4-3 Audio Volume, Brightness and Contrast Controls. 4-1
4-4 Special Display Attributes 4-3
4-5 I/O Instructions 4-3
4-6 Data In A (DIA) Instruction. . . . 4-3
4-7 Data In B (DIB) Instruction. . . . 4-3
4-8 Data In C (DIC) Instruction. . . . 4-3
4-9 Data Out A (DOA) Instruction . . 4-3
4-10 Data Out B (DOB) Instruction Without Start Function. 4-3
4-11 Data Out B (DOB) Instruction With Start Function. 4-3
4-12 Data Out C (DOC) Instruction . . 4-4
4-13 No I/O (NIO) Instruction 4-4
4-14 Read Done (RDDNE) Instruction 4-4
4-15 Read Busy (RDBSY) Instruction 4-4
4-16 Mask Out (MSKO) Instruction . . 4-4

SECTION 5. UNIVERSAL DISK CONTROLLER . . 5-1

5-1 Introduction. 5-1
5-2 I/O Instructions 5-1
5-3 Data Out A (DOA) Instruction . . 5-1
5-4 Data In A (DIA) Instruction - Without Clear 5-3
5-5 Data In A (DIA) Instruction - With Clear 5-3
5-6 Writing Considerations 5-3
5-7 Additional Floppy Considerations 5-3
5-8 Initializing Floppies 5-3

SECTION 6. PROGRAMMABLE COMMUNICATIONS CONTROLLER. 6-1

6-1 Introduction. 6-1
6-2 I/O Instructions. 6-1
6-3 Data Out A (DOA) Instruction . . 6-2
6-4 Data Out B (DOB) Instruction . . 6-4
6-5 Data Out C (DOC) Instruction . . 6-5
6-6 Data In A (DIA) Instruction. . . . 6-6
6-7 Data In B (DIB) Instruction. . . . 6-6
6-8 Data In C (DIC) Instruction. . . . 6-6
6-9 No Input/Output (NIO) Instruction 6-6
6-10 Mask Out (MSKO) Instruction . . 6-6
6-11 Interrupt Processing 6-7
6-12 RBCC8 Instruction 6-7

SECTION 7. AUTO-DIALER CONTROLLER. . . . 7-1

7-1 Introduction. 7-1
7-2 I/O Instructions. 7-1
7-3 Data Out A (DOA) Instruction . . 7-1
7-4 Data In A (DIA) Instruction. . . . 7-1
7-5 Mask Out (MSKO) Instruction . . 7-2
7-6 Interrupt Processing. 7-2
7-7 Programming Rules 7-2

SECTION 8. PRINTER CONTROLLER 8-1

8-1 Introduction. 8-1
8-2 Line Printer I/O Instructions. 8-1
8-3 Data Out A (DOA) Instruction . . 8-1
8-4 Data In A (DIA) Instruction. . . . 8-3
8-5 Mask Out (MSKO) Instruction . . 8-3
8-6 Character Printer I/O Instruction. . . . 8-3
8-7 Data Out A (DOA) Instruction . . 8-3
8-8 Data In A (DIA) Instruction. . . . 8-3
8-9 Mask Out (MSKO) Instruction . . 8-3

TABLE OF CONTENTS (Continued)

SECTION 9. INITIALIZE PROCEDURE........ 9-1

9-1 Introduction.................... 9-1
9-2 APL Switches 9-1
9-3 Initialize Procedure 9-1
9-4 Initialize Routine 9-1
9-5 Looping In Initialize Routine......... 9-2

LIST OF ILLUSTRATIONS

Figure	Title	Page
1-1	J500 Videocomputer	1-1
1-2	J500 Videocomputer Configuration Diagram....................	1-3
2-1	Arithmetic and Logic Units Referenced in Instructions................	2-1
2-2	Instruction Word for Addressing Memory.....................	2-2
2-3	Load and Store Instruction Format	2-4
2-4	Arithmetic Instructions............	2-5
2-5	Logical Instruction Format	2-5
2-6	Skip Instruction Formats...........	2-6
2-7	Transfer-of-Control Instruction Formats....................	2-7
2-8	Shift Instruction Format	2-10
2-9	Register Instruction Formats........	2-13
2-10	Configuration of Status Flags	2-18
4-1	International Keyboard (American)	4-1
8-1	Line Printer Data Character - DOA Instruction................	8-1
8-2	Line Printer DIA Format	8-3
8-3	DOA Word Format	8-4
8-4	Print Command Word Format........	8-4
8-5	Restore Command	8-4
8-6	Carriage Strobe Command..........	8-4
8-7	Paper Feed Command.............	8-4
8-8	Ribbon Lift Command	8-4
8-9	Status Word Format	8-5
9-1	Normal Initialize CRT Display	9-2

LIST OF TABLES

Table	Title	Page
1-1	Functional and Physical Specifications	1-2
1-2	Controls and Switches.............	1-2
2-1	Summary of Addressing Modes.......	2-2
2-2	Notation Used in Instruction Descriptions.................	2-3
2-3	Load and Store Instructions	2-4
2-4	Arithmetic Instructions............	2-5
2-5	Logical Instructions	2-5
2-6	Skip Instructions	2-8
2-7	Transfer-of-Control Instructions	2-8
2-8	Branch-on-Condition Codes	2-10
2-9	Shift Instructions	2-12
2-10	Register Instructions	2-12
2-11	Input/Output Instructions...........	2-15
2-12	Interrupt Priorities	2-16
2-13	Device Addresses................	2-16
2-14	Mask Out Bits	2-18
2-15	Halt and Flag Instructions	2-19
2-16	Status Flags	2-19
2-17	Control Flag Codes...............	2-19
4-1	Eight Bit ASCII Code..............	4-2
5-1	Diskette Initializing Format - Single Density	5-4
5-2	Diskette Initializing Format - Double Density	5-5
8-1	Line Printer Character Set	8-2
8-2	Seven Bit ASCII Code for 96 Character Diablo HyType II and Qume Printers.....................	8-6
8-3	Seven Bit ASCII Code for 88 Character Diablo Word Processing Printer....................	8-7
9-1	DIP Switch Functions - Location 24H...	9-1
A-1	Instruction Set.................	A-1

1-1. INTRODUCTION.

The J500 Videocomputer is a versatile single user word processing and business oriented data processing system. The system is a complete self-contained, stand alone system that uses the same vast store of programs as larger systems currently produced by Jacquard.

The J500 system, Figure 1-1, is packaged in an attractive cabinet designed to compliment and blend with standard office equipment and furniture. Optional peripheral devices are similarly packaged. The J500 and peripheral options have internal power strapping provisions which allow factory set-up for operation on international power standards of 100VAC, 115VAC or 230VAC at 50Hz or 60Hz. Refer to Table 1-1 for functional and physical specifications.

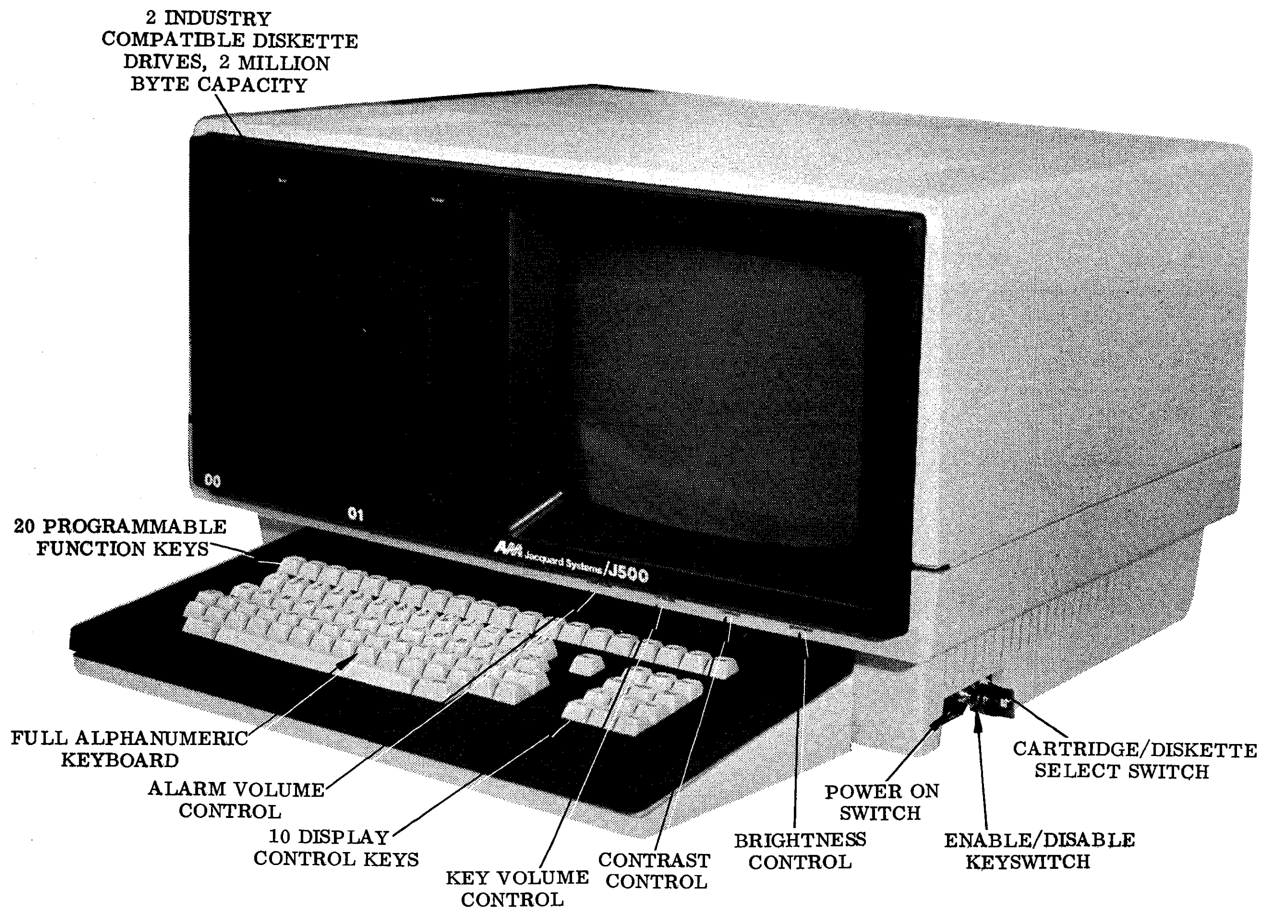


Figure 1-1. J500 Videocomputer

Table 1-1. Functional and Physical Specifications

PROCESSOR

- Word Size - 16 Bits
- Accumulators - Four
- Push Down Stack - 16 Deep
- Status Register
- Program Counter
- Memory Address Register
- Direct Memory Access 1,500,000 bytes/sec.
- Programmed I/O approx. 375 K bytes/sec.
- Interrupt System
- Real Time Clock
- Powerful instruction set

KEYBOARD

- Removable via cable
- Full Upper/Lower ASCII Key Layout
- Display Control Keys (10)
- Edit/Function Keys (20)
- "n" Key Rollover

PHYSICAL

J500 Videocomputer:
Height 14.5 in.
Width 24.5 in.
Depth 17.25 in.
Weight 95 lbs.

Keyboard:
Height 3.5 in.
Width 18.5 in.
Depth 9.25 in.
Weight 2.5 lbs
Total Depth (J500 and Keyboard): 26.5 inches.

CRT DISPLAY

- 12" CRT
- 80 x 24 Format
- High Resolution Characters
- Full ASCII Character Set
- 128 User Programmable Characters
- High Contrast
- Low Reflectance faceplate
- No flicker
- Update at CPU speed

DISKETTES

- 2 Drives Built-in
- Industry compatible format
- 250K, 500K or 1M bytes/Drive
- Random access
- Removable Media

MEMORY

- Semi-Conductor
- 128K bytes (Standard)

ENVIRONMENTAL

Operating Temperature
60° F to 85° F
(15.5° C to 294° C)
ambient air

Humidity: 20% to 80% RH
(non-condensing)

1-2 PHYSICAL/FUNCTIONAL ORGANIZATION.

The J500 consists of:

(1) A 16-bit general purpose digital computer which incorporates state-of-the-art Bi-Polar Proms and discrete TTL logic.

(2) 128K, 8 bit bytes of memory for storage of data and application programs.

(3) Typewriter style keyboard for data entry plus special function keys.

(4) A 12-inch, 1920 character CRT screen for data display.

(5) Two industry compatible diskette drives (floppy disk) for data entry and additional program and data storage.

1-3 PHYSICAL ORGANIZATION.

The design of the J500 Videocomputer simplifies system configuration and I/O Peripheral needs for variety of applications. The central processor, memory, controllers and interfaces are contained on a single 13 1/4" x 25" printed circuit board. This CPU board is mounted in the rear of the J500 chassis and has plug-in interfacing for the available peripheral options. The CRT, Keyboard and the power supply also connect to the CPU board. Operating controls and switches for the J500 are defined in Table 1-2.

The J500 Videocomputer is complete and ready to interface to:

1. Two single or double density floppy disks (internal to J500 cabinet).

2. One 24 line, 80 characters per line CRT display (internal to J500 cabinet).

3. One Printer (external) - either character or line.

4. Two Communication ports of either BISYNC or ASYNC compatible at all standard Baud Rates with complete Modem Controls.

5. An Automatic Dialer.

6. Either 12, or 24 megabyte disk storage with fixed and removable media. Each J500 will support 4-12 M disks or 2-24M disks.

The two Communication Ports provide a completely open ended means of interfacing with Optical Character Readers, Printers, Embossers, etc. In most cases programs are already available for these installations.

Table 1-2. Controls and Switches

Control/Switch	Function
POWER ON Switch	Applies AC line voltage to power supply and internal cooling fans.
ENABLE/ DISABLE Keyswitch	Allows unit to operate in ENABLE position.
CARTRIDGE/ DISKETTE Switch	Device select switch which specifies which device the Auto-Program Load will occur from.
ALARM VOLUME potentiometer	Audible alarm volume control.
KEY VOLUME potentiometer	Keyboard audible "click" volume control.
CONTRAST potentiometer	CRT contrast control.
BRIGHTNESS potentiometer	CRT brightness control.

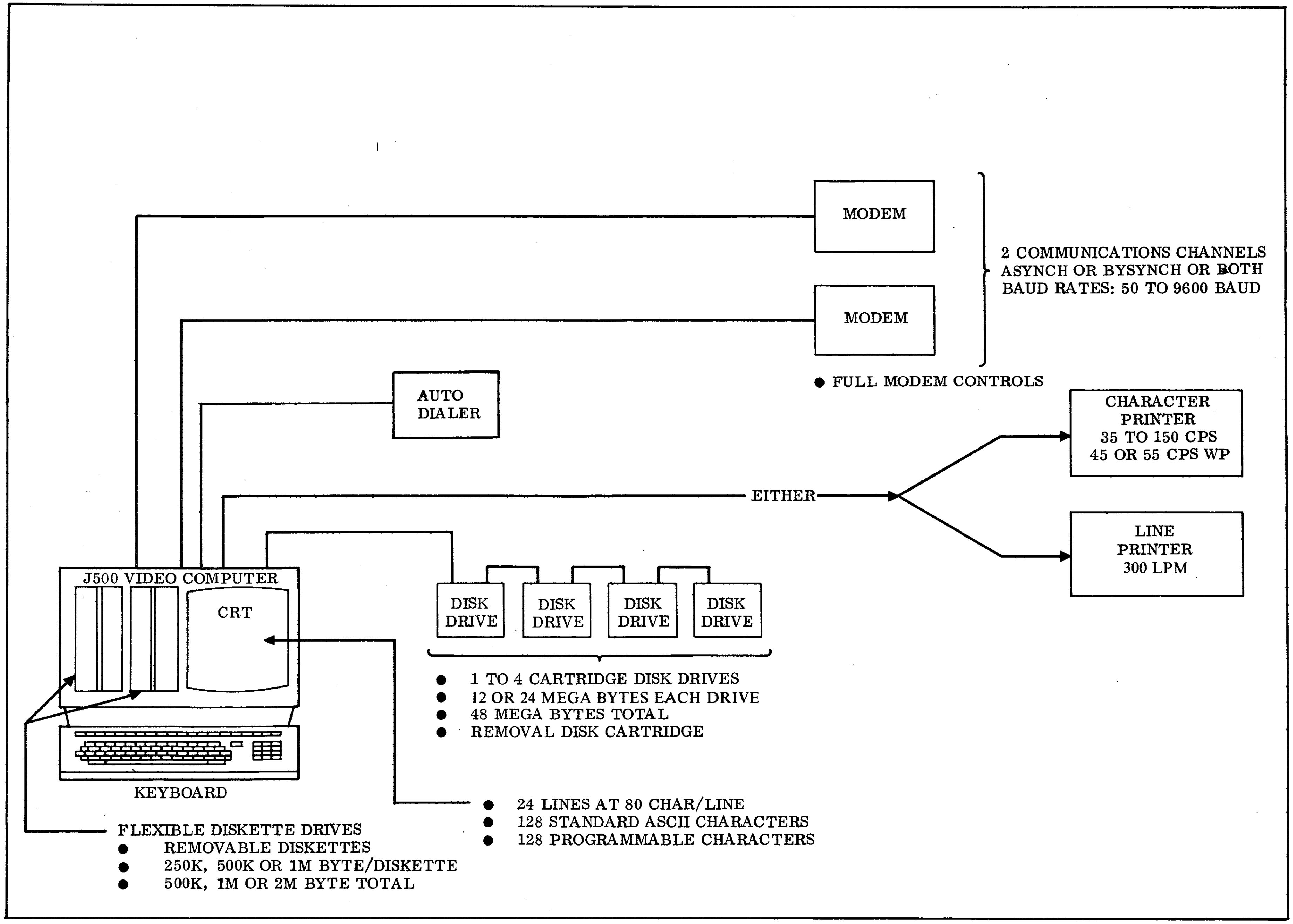

MODEM
MODEM
2 COMMUNICATIONS CHANNELS
ASYNCH OR BYSYNCH OR BOTH
BAUD RATES: 50 TO 9600 BAUD
● FULL MODEM CONTROLS
AUTO DIALER
CHARACTER PRINTER 35 TO 150 CPS 45 OR 55 CPS WP
EITHER
LINE PRINTER 300 LPM
J500 VIDEO COMPUTER
CRT
DISK DRIVE
DISK DRIVE
DISK DRIVE
DISK DRIVE
● 1 TO 4 CARTRIDGE DISK DRIVES
● 12 OR 24 MEGA BYTES EACH DRIVE
● 48 MEGA BYTES TOTAL
● REMOVAL DISK CARTRIDGE
KEYBOARD
● 24 LINES AT 80 CHAR/LINE
● 128 STANDARD ASCII CHARACTERS
● 128 PROGRAMMABLE CHARACTERS
FLEXIBLE DISKETTE DRIVES
● REMOVABLE DISKETTES
● 250K, 500K OR 1M BYTE/DISKETTE
● 500K, 1M OR 2M BYTE TOTAL

Figure 1-2. J500 Videocomputer Configuration Diagram

1-4. FUNCTIONAL ORGANIZATION.

The J500 is a general purpose computer system with a 16-bit word length. The central processor is organized around four accumulators, two of which can be used as index registers. This accumulator/index register organization provides ease in programming and greater efficiency both in time and memory use.

The J500 contains a fast and powerful microprocessor that reads, interprets, and executes MACRO instructions from the J500 main memory. Because of the processor's high speed and powerful micrologic, it is also quite capable of:

1. Controlling the CRT directly
2. Controlling the Floppy Disks including Direct Memory Access
3. Controlling both communication ports
4. Controlling D3000 Disks
5. Controlling the printer
6. Controlling the keyboard
7. Controlling the Auto Dialer

Plus performing lesser tasks such as:

1. Updating the real time clock
2. Managing interrupts
3. Control panel operations
4. Hardware diagnostics for both fault detection and location

The central processor is the control unit for the entire system: it governs all peripheral I/O equipment, performs all arithmetic, logical and data handling operations, and sequences the program. It is connected to the memory by a SUM bus and to the peripheral equipment by a data bus.

The processor handles sixteen bit words that are numbered 0 to 15, left to right. Words are used either as computer instructions in a program, as addresses, or as operands, i.e., dynamic data for processing. The arithmetic instructions operate on fixed point binary numbers, either unsigned or the equivalent signed numbers using the two's complement convention.

1-5. SYSTEM FEATURES.

1-6. EXPANSION CAPABILITIES. The design of the J500 Videocomputer simplifies system configuration and I/O peripheral needs for a variety of applications. Both software and hardware are presently designed to accommodate a combination of printers, communication terminals, disk drives and floppy drives. System expansion is achieved through the cable connectors on the J500 CPU board.

1-7. USER INTERFACE. Data input at the keyboard is entered directly into the processor and displayed on the CRT. Additional data and/or records stored in either the processor's memory or on its diskette can be accessed by the application program. Attachment of a peripheral device such as a printer permits reports to be generated directly or as a result of calculations and data manipulation performed within the processor, after data entry at the keyboard. Thus, the entire data processing operation from data entry to finalized report can be conducted by the user directly and without delay.

1-8. HARDWARE FEATURES.

The J500 has extensive data processing facilities which include: an interrupt system, real time clock and a direct memory access channel.

1-9. DISKETTES. The J500 has two "IBM 3740 format compatible" diskette units. Each diskette unit has a storage capacity of approximately 250K for single-sided single density, 500K for single-sided double-density or 1M bytes for double-sided double density. Programs and/or data may be stored on each of these two diskette units. The J500 has provisions for the addition of two externally mounted diskette units. This increases the on-line data storage capacity for diskettes to approximately four million bytes of data.

Up to four devices can be attached in a daisy chain fashion. All types of diskettes can be mixed within a daisy chain: single-sided single-density, single-sided double-density, and double-sided double-density.

1-10. PERIPHERAL AND COMMUNICATION OPTIONS.

Optional equipment and peripheral and communications interfaces available with the J500 Videocomputer are shown in Figure 1-2. All of the peripheral equipment controllers are contained on the J500 CPU board.

OPERATING PROCEDURES

Refer to Section 9 for information on power turn-on procedures and initialization of the J500.

2-1. INTRODUCTION.

The J500 Instruction Set consists of eight functional classes of instructions:

- LOAD AND STORE
- ARITHMETIC
- LOGICAL
- SKIP
- SHIFTS
- TRANSFER OF CONTROL
- REGISTER
- INPUT/OUTPUT AND MISCELLANEOUS

The instructions in each functional class are described as a group. The description of each instruction includes the name of the instruction, its mnemonic, its word format, its operation in the form of an equation, and an explanation of its operation. A tabulated summary of each type of instruction precedes the detailed descriptions. Brief descriptions of the registers referred to in the instruction descriptions are also provided.

2-2. ARITHMETIC AND LOGIC UNITS REFERENCED IN INSTRUCTIONS (See Figure 2-1).

- PUSH-DOWN STACK
- STATUS REGISTER
- PROGRAM COUNTER (PC)
- ACCUMULATOR 0 (AC0)
- ACCUMULATOR 1 (AC1)
- ACCUMULATOR 2 (AC2)
- ACCUMULATOR 3 (AC3)

2-3. PUSH-DOWN STACK (LIFO). The microprocessor has a hardware stack that data may be stored in or retrieved from on a last-in/first-out basis. The stack is 16 words deep and is accessible through the top location. As a data word is entered into the stack, the contents of the top location and each other location are pushed downward to the next lower level; if the stack is full, the word in the bottom location is lost. Conversely, the contents of the top location are pulled from the stack during retrieval of a data word; the top location and each lower location are replaced by the contents of the next lower location, and zeroes are entered into the bottom location.

The stack is used primarily for saving status during interrupts and for saving subroutine return addresses. It may be used also for temporary storage of data using the PUSH, PULL, XCHRS, PUSHF, and PULLF instructions (described later in this section).

2-4. STATUS REGISTER. There are 16 status flags in the status register. These flags may be pushed onto the stack (saved) or may be loaded from the stack (restored). During such operations, the flags are configured as a 16-bit word: The LNK (Link), OV (Overflow), and CY (Carry) flags are the first, second, and third most significant bits,

- PUSH DOWN STACK
- STATUS REGISTER
- PROGRAM COUNTER (PC)
- ACCUMULATOR 0 (AC0)
- ACCUMULATOR 1 (AC1)
- ACCUMULATOR 2 (AC2)
- ACCUMULATOR 3 (AC3)

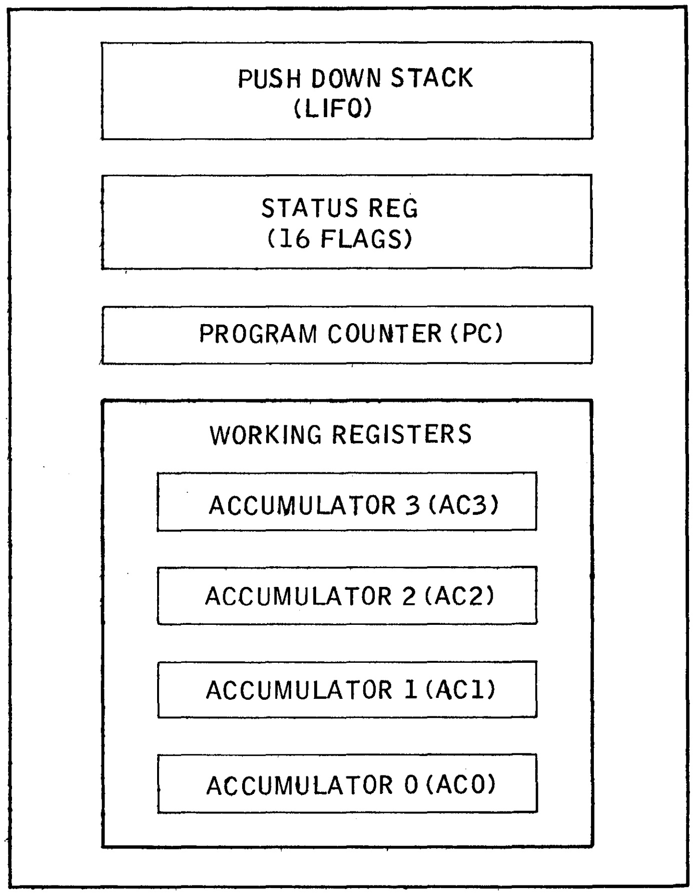


Figure 2-1. Arithmetic and Logic Units Referenced in Instructions

respectively, and the remaining 13 general-purpose flags comprise the remaining 13 less significant bits (Figure 2-11). Note that the SELECT flag is not included in the status register.

The LNK flag is primarily used in some shifting operations, and the CY and OV flags are adjuncts for arithmetic operations. The specific uses of the flags are elaborated upon in the appropriate instruction descriptions.

2-5. PROGRAM COUNTER (PC). The Program Counter (PC) holds the address of the next instruction to be executed. When there is a branch to another address in the main memory, the branch address is set into the Program Counter. A skip instruction merely increments the Program Counter by 1, thus causing one instruction to be skipped.

2-6. ACCUMULATORS 0, 1, 2, AND 3 (AC0, AC1, AC2, AND AC3). The accumulators are used as working registers for data manipulation. Data may be fetched from a location in memory to an accumulator, and may be stored from an accumulator to a location in memory. The particular accumulator to take part in an operation is specified by the programmer in the appropriate instruction.

2-7. DATA AND INSTRUCTIONS.

2-8. DATA REPRESENTATION. Data is represented in the microprocessor in twos-complement integer notation. In this system, the negative of a number is formed by complementing each bit in the data word and adding 1 to the complemented number. The sign is indicated by the most significant bit. In the 16-bit word of the CPU, when bit "0" is a "0", it denotes a positive number; when it is a "1", it denotes a negative number. Maximum number ranges for this system are $7FFF_{16}$ ($+32767_{10}$) and 8000_{16} (-32768_{10}). The carry flag will be set if there is a carry out of bit 0 as the result of an addition operation. The overflow flag will be set if bit 0 of the result of an addition is different than the signs of the operands. Overflow cannot occur if the operands have different signs. Since subtraction is performed by adding the 2's complement of the subtrahend to the minuend, these rules apply to subtraction as well.

2-9. INSTRUCTIONS. There are eight classes of instructions. Each class of instruction and the associated instructions are summarized in a table preceding the descriptions of the instructions. Also, the applicable instruction word formats are defined.

2-10. MEMORY ADDRESSING.

The instruction set provides for direct and indirect memory addressing. For direct addressing, three distinct modes are available: base page (or absolute), program-counter relative, and indexed. The mode of addressing is specified by the XR field of the simplified instruction word format shown in Figure 2-2.

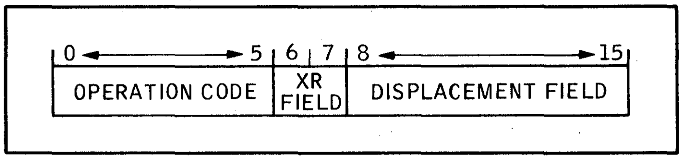


Figure 2-2. Instruction Word for Addressing Memory

2-11. BASE PAGE ADDRESSING. When the XR field is 00, it specifies base page addressing. Base page is directly accessible from any location in the address space of the memory. In this mode, the effective address is formed by setting bits 0 through 7 to zero and using the value of the 8-bit displacement field as an absolute address. Up to 256 words (locations 0 through 255) may be addressed in this way.

2-12. PROGRAM-COUNTER RELATIVE ADDRESSING. Program-counter relative addressing is specified when the XR field is 01. The displacement is treated as a signed number such that its sign bit (bit 8) is propagated to bits 0 through 7, and the effective address is formed by adding the contents of the PC to the resulting number. This permits PC-relative addressing -128 and +127 locations from the PC value; however, at the time of formation of the address, the PC has already been incremented in the microprogram and is pointing to the next macroinstruction. Because of this, the actual addressing range is from -127 to +128 from the current instruction.

2-13. INDEXED ADDRESSING. Indexed addressing is done with reference to only Accumulator 2 or 3 (AC2 or AC3). In this mode, the displacement field is again interpreted as a signed 8-bit number from -128 to +127 with the sign (bit 8) extended through bits 0 through 7. The contents of the chosen index register (AC2 when $XR = 10_2$ and AC3 when $XR = 11_2$) are added to the number formed from the displacement value to yield an effective address that can reach any location in 65,536 words of memory. A summary of addressing modes is shown in Table 2-1.

Table 2-1. Summary of Addressing Modes

XR FIELD	ADDRESSING MODE	EFFECTIVE ADDRESS	RANGE
00	Base	EA = disp	$0 \le disp \le 255$
01	Relative to Program Counter	EA = disp + (PC)	$-128 \le disp \le 127$
10	Relative to AC2	EA = disp + (AC2)	$-128 \le disp \le 127$
11	Relative to AC3	EA = disp + (AC3)	$-128 \le disp \le 127$

2-14. INDIRECT ADDRESSING. Indirect addressing is accomplished by first calculating the effective address (EA) using the same method used for direct addressing; the memory location at this address contains a number that is then used as the address of the operand. Only the following instructions use indirect addressing:

- Load Indirect (See Table 2-3)
- Store Indirect (See Table 2-3)
- Jump Indirect (See Table 2-7)
- Jump to Subroutine Indirect (See Table 2-7)

2-15. NOTATIONS AND SYMBOLS USED IN MICROPROCESSOR INSTRUCTION DESCRIPTIONS.

Definitions of the notations and symbols used in the J500 descriptions are listed in Table 2-2. The notations are listed in alphabetical order followed by the symbols. Upper-case mnemonics refer to fields in the instruction word; lower-case mnemonics refer to the numerical value of the corresponding fields. In cases where both lower- and upper-case mnemonics are composed of the same letters, only the lower-case mnemonic is shown in Table 2-2. The use of a lower-case notation designates variables.

Table 2-2. Notation Used In Instruction Descriptions

NOTATION	MEANING
ACr	Denotes a specific working register (AC0, AC1, AC2, or AC3), where r is the number of the accumulator referenced in the instruction.
AR	Denotes the address register used for addressing memory or peripheral devices.
cc	Denotes the 4-bit condition code value for conditional branch instructions.
ctl	Denotes the 7-bit control-field value for flag, input/output, and miscellaneous instructions.
CY	Indicates that the Carry flag is set if there is a carry due to the instruction (either an addition or a subtraction).
disp	Stands for displacement value and it represents an operand in a non-memory reference instruction or an address field in a memory reference instruction. It is an 8-bit, signed twos-complement number except when base page is referenced; in the latter case, it is unsigned.
dr	Denotes the number of a destination working register that is specified in the instruction-word field. The working register is limited to one of four: AC0, AC1, AC2, or AC3.
EA	Denotes the effective address specified by the instruction directly, indirectly, or by indexing. The contents of the effective address are used during execution of an instruction. (See Table 2-1).
fc	Denotes the number of the referenced flag (See Table 2-13).
INT EN	Denotes the Interrupt Enable Control Flag.
IOREG	Denotes an input/output register in a peripheral device.
L	Denotes 1-bit link (L) flag.
OV	Indicates that the overflow flag is set if there is an overflow due to the instruction (either an addition or a subtraction). (See Par. 2-8)
PC	Denotes the program counter. During address formation, it is incremented by 1 to contain an address 1 greater than that of the instruction being executed.
r	Denotes the number of a working register that is specified in the instruction-word field. The working register is limited to one of four: AC0, AC1, AC2, or AC3.
SEL	Denotes the Select control flag. It is used to select the carry or overflow for sensing with the Branch-On Condition instruction, and to include the link bit (L) in shift operations. Note that this flag is not included in the Status flag register (Figure 2-11).
sr	Denotes the number of a source working register that is specified in the instruction-word field. The working register is limited to one of four: AC0, AC1, AC2, or AC3.
STK	Denotes the hardware Push-Down Stack that data is stored in or retrieved from.
xr	When not zero, this value designates the register to be used in the indexed and relative memory-addressing modes. (See Table 2-1).
( )	Denotes the contents of the item within the parentheses. (ACr) is read as "the contents of ACr." (EA) is read as "the contents of EA."
[ ]	Denotes "the result of."
~	Indicates the logical complement (ones complement) of the value on the right-hand side of ~.
$\rightarrow$	Means "replaces."
$\leftarrow$	Means "is replaced by."
@	Appearing in the operand field of an instruction, denotes indirect addressing.
$\cdot$ $\wedge$	Denotes an AND operation (either).
$\vee$	Denotes an OR operation.
$\veebar$	Denotes an exclusive OR operation.

2-16. INSTRUCTION DESCRIPTIONS.

Each class and subclass of instruction is introduced by a table that lists and summarizes the instructions and then the word format is illustrated. The detailed descriptions provide the following information:

- Name of instruction followed by operation code mnemonic in parentheses
- Operation Code in word format diagram
- Operation in equation notation
- Detailed description of operation

2-17. LOAD AND STORE INSTRUCTIONS.

The four instructions in this group are summarized in Table 2-3, then individually described. The word format is shown in Figure 2-3.

Table 2-3. Load and Store Instructions

INSTRUCTION	OP CODE	OPERATION	ASSEMBLER FORMAT
LOAD	1000	$(ACr) \leftarrow (EA)$	LD r, disp(xr)
LOAD INDIRECT	1001	$(ACr) \leftarrow ((EA))$	LD r, @disp(xr)
STORE	1010	$(EA) \leftarrow (ACr)$	ST r, disp(xr)
STORE INDIRECT	1011	$((EA)) \leftarrow (ACr)$	ST r, @disp(xr)

NOTE

For indirect operations, the symbol @ must precede the memory location designated in the operand field of the assembler instruction.

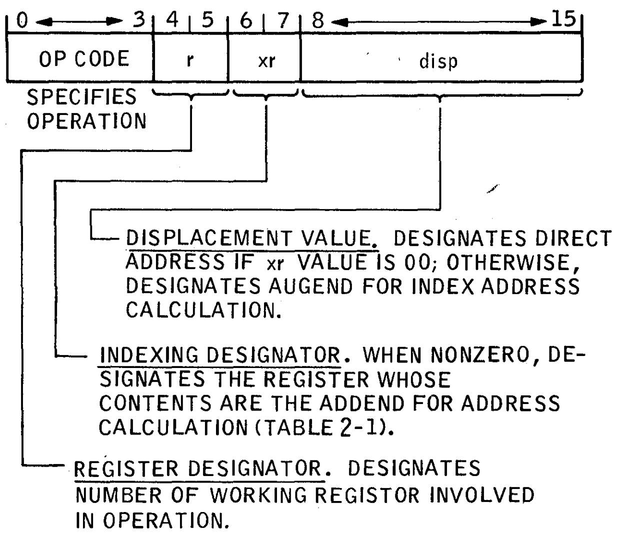

0 3 4 5 6 7 8 15
OP CODE
r
xr
disp
SPECIFIES OPERATION
DISPLACEMENT VALUE. DESIGNATES DIRECT ADDRESS IF xr VALUE IS 00; OTHERWISE, DESIGNATES AUGEND FOR INDEX ADDRESS CALCULATION.
INDEXING DESIGNATOR. WHEN NONZERO, DESIGNATES THE REGISTER WHOSE CONTENTS ARE THE ADDEND FOR ADDRESS CALCULATION (TABLE 2-1).
REGISTER DESIGNATOR. DESIGNATES NUMBER OF WORKING REGISTOR INVOLVED IN OPERATION.

Figure 2-3. Load and Store Instruction Format

LOAD (LD) 8000_{16}

0 – 3	4 5	6 7	8 – 15
1 0 0 0	r	xr	disp

Operation: $(ACr) \leftarrow (EA)$

Description: The contents of ACr are replaced by the contents of EA. The initial contents of ACr are lost; the contents of EA are unaltered.

LOAD INDIRECT (LD@) 9000_{16}

0 – 3	4 5	6 7	8 – 15
1 0 0 1	r	xr	disp

Operation: $(ACr) \leftarrow ((EA))$

Description: The contents of ACr are replaced indirectly by the contents of EA. The initial contents of ACr are lost; the contents of EA and the location that designates EA are unaltered.

STORE (ST) $A000_{16}$

0 – 3	4 5	6 7	8 – 15
1 0 1 0	r	xr	disp

Operation: $(EA) \leftarrow (ACr)$

Description: The contents of EA are replaced by the contents of ACr. The initial contents of EA are lost; the contents of ACr are unaltered.

STORE INDIRECT (ST@) $B000_{16}$

0 – 3	4 5	6 7	8 – 15
1 0 1 1	r	xr	disp

Operation: $((EA)) \leftarrow (ACr)$

Description: The contents of EA are replaced indirectly by ACr. The initial contents of EA are lost; the contents of ACr and the location that designates EA are unaltered.

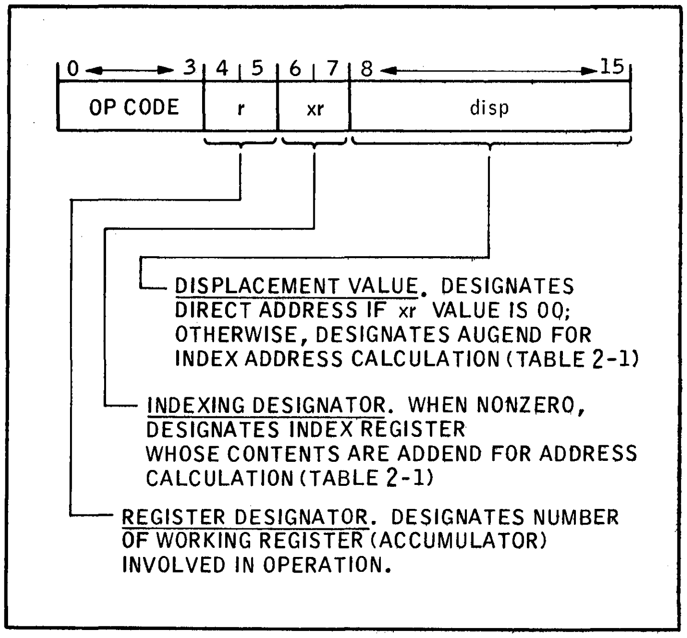
Programming Note: Since the J500 fetches the "next" instruction from memory while executing an instruction, it is incorrect to store into the "next" location with the expectation that the value stored will then be accessed for execution. Storage into a location which will be executed "after next" may also be unsafe.

2-18. ARITHMETIC INSTRUCTIONS.

The two instructions in this group are summarized in Table 2-4 and then described individually. Either of these instructions may be carried out with any of the four general-purpose accumulators (AC0, 1, 2, or 3). The word format is shown in Figure 2-4.

Table 2-4. Arithmetic Instructions

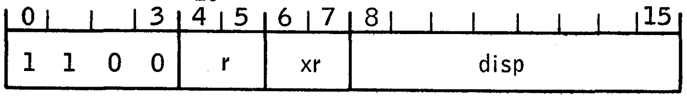
INSTRUCTION	OPCODE	OPERATION	ASSEMBLER FORMAT
ADD (ADD)	1100	$(ACr) \leftarrow (ACr) + (EA), OV, CY$	ADD r, disp(xr)
SUBTRACT (SUB)	1101	$(ACr) \leftarrow (ACr) + \sim(EA) + 1, OV, CY$	SUB r, disp(xr)



0 3 4 5 6 7 8 15
OP CODE
r
xr
disp
DISPLACEMENT VALUE. DESIGNATES DIRECT ADDRESS IF xr VALUE IS 00; OTHERWISE, DESIGNATES AUGEND FOR INDEX ADDRESS CALCULATION (TABLE 2-1)
INDEXING DESIGNATOR. WHEN NONZERO, DESIGNATES INDEX REGISTER WHOSE CONTENTS ARE ADDEND FOR ADDRESS CALCULATION (TABLE 2-1)
REGISTER DESIGNATOR. DESIGNATES NUMBER OF WORKING REGISTER (ACCUMULATOR) INVOLVED IN OPERATION.

Figure 2-4. Arithmetic Instructions

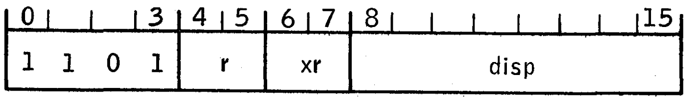
ADD (ADD) $C000_{16}$



Operation: $(ACr) \leftarrow (ACr) + (EA)$, OV, CY

Description: The contents of ACr are added algebraically to the contents of the effective memory location EA. The sum is stored in ACr, and the contents of EA are unaltered. The preceding contents of ACr are lost. The carry and overflow flags are set according to the result of the operation. (Refer to para. 2-8)

SUBTRACT (SUB) $D000_{16}$



Operation: $(ACr) \leftarrow (ACr) + \sim(EA) + 1$, OV CY

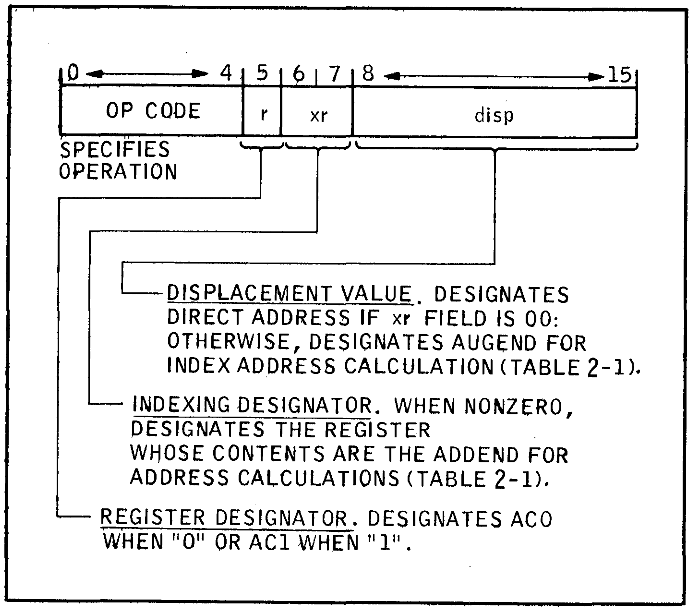
Description: The contents of ACr are added to the twos complement of the effective memory location EA. The result is stored in ACr, and the effective memory location is unaltered. The carry and overflow flags are set according to the result of the operation. (Refer to para. 2-8.)

2-19. LOGICAL INSTRUCTIONS.

There are two instructions in this group, summarized in Table 2-5 and then described individually. Either of the instructions may be carried out with only two of the general-purpose accumulators, AC0 or AC1. The word format is shown in Figure 2-5.

Table 2-5. Logical Instructions

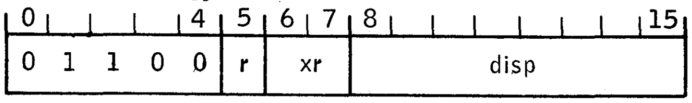
INSTRUCTION	OPCODE	OPERATION	ASSEMBLER FORMAT
AND	01100	$(ACr) \leftarrow (ACr) \wedge (EA)$	AND r, disp(xr)
OR	01101	$(ACr) \leftarrow (ACr) \vee (EA)$	OR r, disp(xr)



0 4 5 6 7 8 15
OP CODE
r
xr
disp
SPECIFIES OPERATION
DISPLACEMENT VALUE. DESIGNATES DIRECT ADDRESS IF xr FIELD IS 00: OTHERWISE, DESIGNATES AUGEND FOR INDEX ADDRESS CALCULATION (TABLE 2-1).
INDEXING DESIGNATOR. WHEN NONZERO, DESIGNATES THE REGISTER WHOSE CONTENTS ARE THE ADDEND FOR ADDRESS CALCULATIONS (TABLE 2-1).
REGISTER DESIGNATOR. DESIGNATES AC0 WHEN "0" OR AC1 WHEN "1".

Figure 2-5. Logical Instruction Format

AND (AND) 6000_{16}



Operation: $(ACr) \leftarrow (ACr) \wedge (EA)$

Description: The contents of ACr (where r is either 0 or 1) and the contents of the effective memory location EA are ANDed, and the result is stored in ACr. The initial contents of ACr are lost, and the contents of EA are unaltered.

OR (OR) 6800_{16}

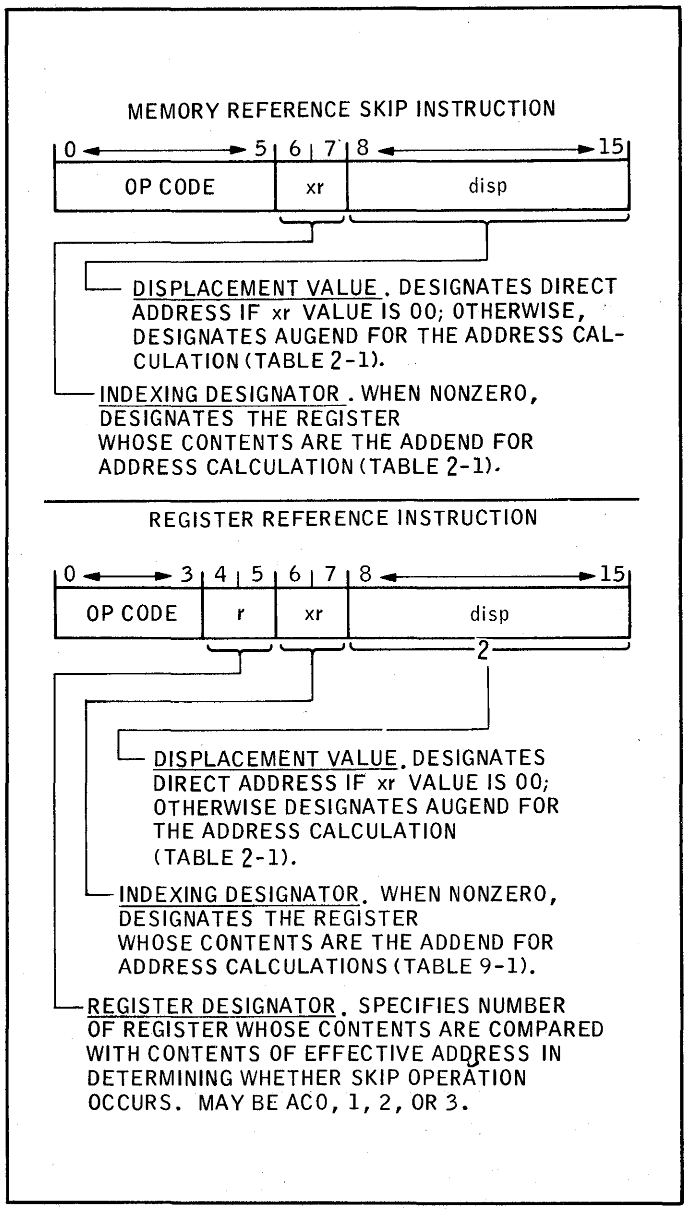
0				4	5	6	7	8							15
0	1	1	0	1	r	xr		disp							

Operation: (ACr) ← (ACr) ∨ (EA)

Description: The contents of ACr (where r is either 0 or 1) and the contents of the effective memory location EA are ORed inclusively, and the result is stored in ACr. The initial contents of ACr are lost, and the contents of EA are unaltered.

2-20. SKIP INSTRUCTIONS.

The five skip instructions are summarized in Table 2-6. The three word formats required are shown in Figure 2-6.



MEMORY REFERENCE SKIP INSTRUCTION

0 → 5	6 7	8 → 15
OP CODE	xr	disp

DISPLACEMENT VALUE. DESIGNATES DIRECT ADDRESS IF xr VALUE IS 00; OTHERWISE, DESIGNATES AUGEND FOR THE ADDRESS CALCULATION (TABLE 2-1).

INDEXING DESIGNATOR. WHEN NONZERO, DESIGNATES THE REGISTER WHOSE CONTENTS ARE THE ADDEND FOR ADDRESS CALCULATION (TABLE 2-1).

REGISTER REFERENCE INSTRUCTION

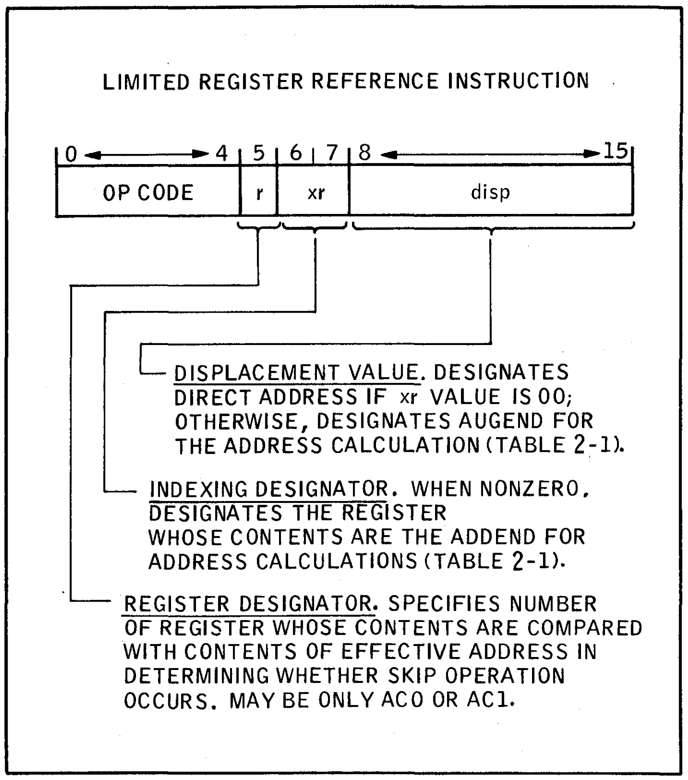
0 → 3	4 5	6 7	8 → 15
OP CODE	r	xr	disp

DISPLACEMENT VALUE. DESIGNATES DIRECT ADDRESS IF xr VALUE IS 00; OTHERWISE DESIGNATES AUGEND FOR THE ADDRESS CALCULATION (TABLE 2-1).

INDEXING DESIGNATOR. WHEN NONZERO, DESIGNATES THE REGISTER WHOSE CONTENTS ARE THE ADDEND FOR ADDRESS CALCULATIONS (TABLE 9-1).

REGISTER DESIGNATOR. SPECIFIES NUMBER OF REGISTER WHOSE CONTENTS ARE COMPARED WITH CONTENTS OF EFFECTIVE ADDRESS IN DETERMINING WHETHER SKIP OPERATION OCCURS. MAY BE AC0, 1, 2, OR 3.

Figure 2-6. Skip Instruction Formats



LIMITED REGISTER REFERENCE INSTRUCTION

0 → 4	5	6 7	8 → 15
OP CODE	r	xr	disp

DISPLACEMENT VALUE. DESIGNATES DIRECT ADDRESS IF xr VALUE IS 00; OTHERWISE, DESIGNATES AUGEND FOR THE ADDRESS CALCULATION (TABLE 2-1).

INDEXING DESIGNATOR. WHEN NONZERO, DESIGNATES THE REGISTER WHOSE CONTENTS ARE THE ADDEND FOR ADDRESS CALCULATIONS (TABLE 2-1).

REGISTER DESIGNATOR. SPECIFIES NUMBER OF REGISTER WHOSE CONTENTS ARE COMPARED WITH CONTENTS OF EFFECTIVE ADDRESS IN DETERMINING WHETHER SKIP OPERATION OCCURS. MAY BE ONLY AC0 OR AC1.

Figure 2-6. Skip Instruction Formats (Continued)

INCREMENT AND SKIP IF ZERO (ISZ) 7800_{16}

0					5	6	7	8							15
0	1	1	1	1	0	xr		disp							

Operation: (EA) ← (EA) + 1; if (EA) = 0, (PC) ← (PC) + 1

Description: The contents of EA are incremented by 1. The new contents of EA are tested to determine whether they equal zero. If the new contents of EA equal zero, the contents of PC are incremented by 1, thus skipping the memory location designated by the initial contents of PC.

DECREMENT AND SKIP IF ZERO (DSZ) $7C00_{16}$

0					5	6	7	8							15
0	1	1	1	1	1	xr		disp							

Operation: (EA) ← (EA) - 1; if (EA) = 0, (PC) ← (PC) + 1

Description: The contents of EA are decremented by 1. The new contents of EA are tested to determine whether they equal zero. If the new contents of EA equal zero, the contents of PC are incremented by 1, thus skipping the memory location designated by the initial contents of PC.

SKIP IF GREATER (SKG) $E000_{16}$

0			3	4	5	6	7	8							15
1	1	1	0	r		xr		disp							

Operation: If (ACr) > (EA), (PC)◄(PC) + 1

Description: The contents of ACr (when r is AC0, 1, 2, or 3) and the contents of the effective memory location EA are compared on an algebraic basis with due regard to the signs of the two operands. If the contents of ACr are greater than the contents of EA, the contents of PC are incremented by 1, thus skipping the memory location designated by the initial contents of PC.

Programming note: Care must be exercised when using the SKG instruction to compare memory addresses since the address in the upper half of address space (8000_{16}: $FFFF_{16}$) will compare less than the address in the lower half ($0:7FFF_{16}$) due to the arithemetic rather than logical compare which is employed in the CPU.

SKIP IF NOT EQUAL (SKNE) $F000_{16}$

0			3	4	5	6	7	8							15
1	1	1	1	r		xr		disp							

Operation: If ACr $\neq$ (EA), (PC)◄(PC) + 1

Description: The contents of ACr (where ACr is AC0 1, 2, or 3) and the contents of the effective memory location EA are compared. If the contents of ACr and the effective memory location EA are not equal, the contents of PC are incremented, thus skipping the memory location designated by the initial contents of PC.

SKIP IF AND IS ZERO (SKAZ) 7000_{16}

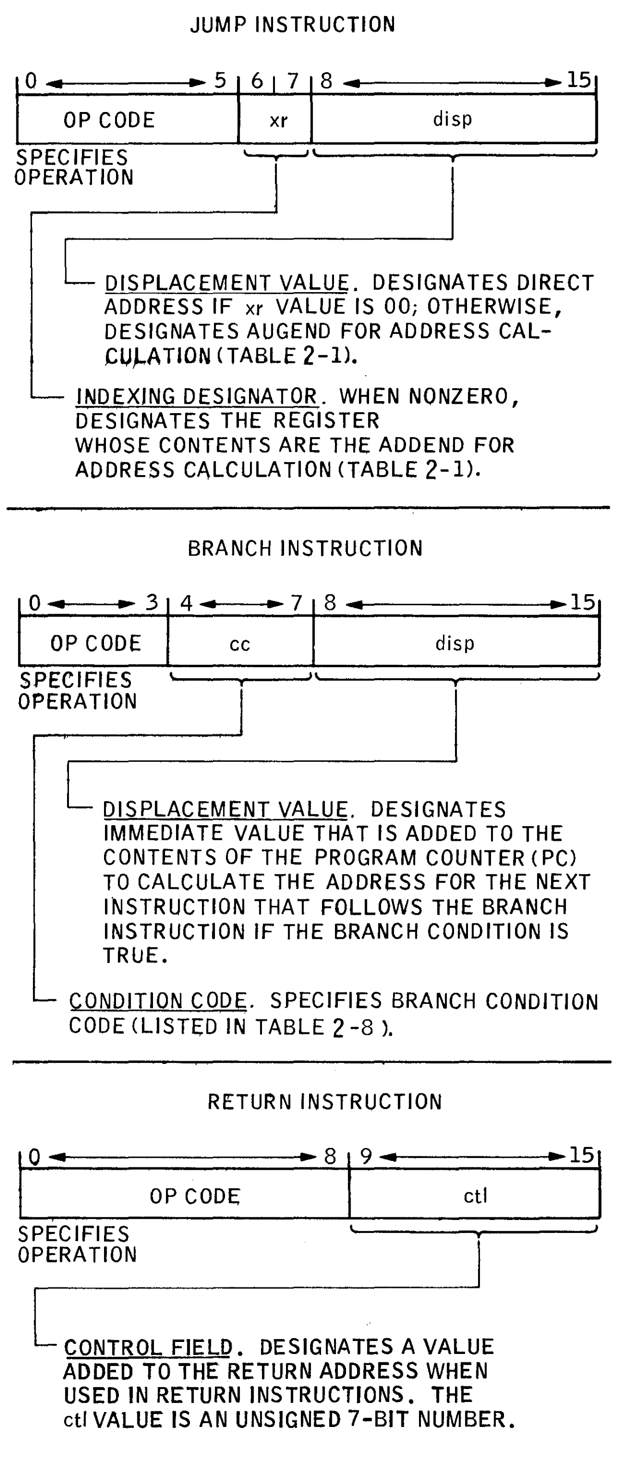
0				4	5	6	7	8							15
0	1	1	1	0	r	xr		disp							

Operation: If $[(ACr) \wedge (EA)] = 0$, (PC)◄(PC) + 1

Description: The contents of ACr (where r is either 0 or 1) and the contents of the effective memory location EA are ANDed. If the result equals zero, the contents of PC are incremented by 1, thus skipping the instruction designated by the initial contents of PC. Note that the contents of the register r is not altered by this instruction.

2-21. TRANSFER-OF-CONTROL INSTRUCTIONS.

The eight instructions in this group are summerized in Table 2-7. The three wordformats required are shown in Figure 2-7.



DISPLACEMENT VALUE. DESIGNATES DIRECT ADDRESS IF xr VALUE IS 00; OTHERWISE, DESIGNATES AUGEND FOR ADDRESS CALCULATION (TABLE 2-1).

INDEXING DESIGNATOR. WHEN NONZERO, DESIGNATES THE REGISTER WHOSE CONTENTS ARE THE ADDEND FOR ADDRESS CALCULATION (TABLE 2-1).

DISPLACEMENT VALUE. DESIGNATES IMMEDIATE VALUE THAT IS ADDED TO THE CONTENTS OF THE PROGRAM COUNTER (PC) TO CALCULATE THE ADDRESS FOR THE NEXT INSTRUCTION THAT FOLLOWS THE BRANCH INSTRUCTION IF THE BRANCH CONDITION IS TRUE.

CONDITION CODE. SPECIFIES BRANCH CONDITION CODE (LISTED IN TABLE 2-8).

CONTROL FIELD. DESIGNATES A VALUE ADDED TO THE RETURN ADDRESS WHEN USED IN RETURN INSTRUCTIONS. THE ctl VALUE IS AN UNSIGNED 7-BIT NUMBER.

Figure 2-7. Transfer-of-Control Instruction Formats

Table 2-6. Skip Instructions

INSTRUCTION	OPERATION CODE	OPERATION	ASSEMBLER FORMAT
Memory References			
INCREMENT AND SKIP IF ZERO	011110	(EA) $\leftarrow$ (EA) + 1; IF (EA) = 0, (PC) $\leftarrow$ (PC) + 1	ISZ disp (xr)
DECREMENT AND SKIP IF ZERO	011111	(EA) $\leftarrow$ (EA) - 1; IF (EA) = 0, (PC) $\leftarrow$ (PC) + 1	DSZ disp (xr)
Register References			
SKIP IF GREATER	1110	IF (ACr) > (EA), (PC) $\leftarrow$ (PC) + 1	SKG r, disp (xr)
SKIP IF NOT EQUAL	1111	IF (ACr) $\neq$ (EA), (PC) $\leftarrow$ (PC) + 1	SKNE r, disp (xr)
Limited Register Reference			
SKIP IF AND IS ZERO	01110	IF $[(ACr) \wedge (EA)] = 0$, (PC) $\leftarrow$ (PC) + 1	SKAZ r, disp (xr)

Table 2-7. Transfer-of-Control Instructions

INSTRUCTION	OPERATION CODE	OPERATION	ASSEMBLER FORMAT
Jumps			
JUMP	001000	(PC) $\leftarrow$ EA	JMP disp(xr)
JUMP INDIRECT	001001	(PC) $\leftarrow$ (EA)	JMP@ disp(xr)
JUMP TO SUBROUTINE	001010	(STK) $\leftarrow$ (PC); (PC) $\leftarrow$ EA	JSR disp(xr)
JUMP TO SUBROUTINE INDIRECT	001011	(STK) $\leftarrow$ (PC); (PC) $\leftarrow$ (EA)	JSR@ disp(xr)
Branch			
BRANCH-ON CONDITION	0001	If CC IS TRUE (PC) $\leftarrow$ (PC) + disp	BOC cc, disp
Returns			
RETURN FROM INTERRUPT	000000010	(PC) $\leftarrow$ (STK) + ctl; INTEN FLAG SET	RTI ctl
RETURN FROM SUBROUTINE	000000100	(PC) $\leftarrow$ (STK) + ctl	RTS ctl

JUMP (JMP) 2000_{16}

0 … 5	6 7	8 … 15
0 0 1 0 0 0	xr	disp

Operation: (PC) ◄ EA

Description: The effective address EA replaces the contents of PC. The next instruction is fetched from the location designated by the new contents of PC.

JUMP (JMP@) INDIRECT 2400_{16}

0 … 5	6 7	8 … 15
0 0 1 0 0 1	xr	disp

Operation: (PC) ◄ (EA)

Description: The contents of the effective address (EA) replaces the contents of PC. The next instruction is fetched from the location designated by the new contents of PC.

JUMP TO SUBROUTINE (JSR) 2800_{16}

0 … 5	6 7	8 … 15
0 0 1 0 1 0	xr	disp

Operation: (STK) ◄ (PC), (PC) ◄ (EA)

Description: The contents of PC are stored in the top of the stack. The effective address EA replaces the contents of PC. The next instruction is fetched from the location designated by the new contents of PC.

Programming Note: Subroutine jumps utilize stack locations and a series of JSR's without intervening returns (see Return From Subroutine) may cause a stack overflow. Jacquard's System II Operating System provides user services which require some stack locations. It is advisable to limit the depth of subroutine calls within user programs.

JUMP TO SUBROUTINE INDIRECT (JSR@) $2C00_{16}$

0 … 5	6 7	8 … 15
0 0 1 0 1 1	xr	disp

Operation: (STK) ◄ (PC) (PC) ◄ (EA)

Description: The contents of PC are stored in the top of the stack. The contents of the effective address (EA) replace the contents of PC. The next instruction is fetched from the location designated by the new contents of PC.

BRANCH-ON CONDITION (BOC) 1000_{16}

0 … 3	4 … 7	8 … 15
0 0 0 1	cc	disp

Operation: (PC) ◄ (PC) + disp (sign extended from bit 8 through bit 0)

Description: There are 16 possible condition codes (cc). These are listed in Table 2-8. If the condition for branching designated by cc is true, the value of disp (sign extended from bit 8 through bit Ø) is added to the contents of PC, and the sum is stored in PC. The initial contents of PC are lost. Program control is transferred to the location specified by the new contents of PC.

NOTES

(1) PC is always incremented by 1 immediately following the fetching of an instruction, so the contents of PC during execution of an instruction is 1 greater than the address of that instruction. This must be considered during execution of the BOC instruction: for example, if the address of the BOC instruction is 100, then 101 is added to the value of disp (sign extended).

(2) The disp field is a signed 8-bit number, whose sign is extended from bit 8 through bit Ø to form a 16-bit number (including sign). Thus, the range of addressing with a BOC instruction is -127 to +128 relative to the address of the current instruction.

(3) Alternate mnemonics are offered in the System II assembler for certain commonly used condition codes. The assembler will recognize the BOC operation code along with a condition code value in the operand field. As an alternate the following table lists certain op-codes which imply a condition code; the explicit condition code value must be omitted when using op-codes from this table.

Op-Code	Equivalent		Condition
BZR	BOC	1	AC0 = 0
BGE	BOC	2	AC0 > 0
BB15	BOC	3	AC0 bit 15 = 1
BB14	BOC	4	AC0 bit 14 = 1
BNZ	BOC	5	AC0 ≠ 0
BCYOF	BOC	10	CY/OV = 1 (per SELECT flag)
BLE	BOC	11	AC0 ≤ 0
BIOBS	BOC	12	I/O busy = true
BIODS	BOC	13	I/O done = true

Table 2-8. Branch-On-Condition Codes

CONDITION CODE	CONDITION TESTED	ALTERNATE OP-CODE	REMARKS
0000			Undefined
0001	(AC0) = 0	BZR	
0010	(AC0) ≥ 0	BGE	
0011	Bit 15 of AC0 = 1	BB15	
0100	Bit 14 of AC0 = 1	BB14	
0101	(AC0) ≠ 0	BNZ	
0110			Undefined
0111			Undefined
1000	STACK FULL FLAG	BOCSFL	
1001	INTERRUPT ENABLE FLAG	INTEN	
1010	CARRY/ OVERFLOW = 1	BCYOF	Carry if SEL = 0; overflow if SEL = 1
1011	(AC0) ≤ 0	BLE	
1100	I/O BUSY FLAG = 1	BIOBS	
1101	I/O DONE FLAG = 1	BIODS	
1110	ZERO		
1111	STACK FULL FLAG	BOCSFL	

RETURN FROM INTERRUPT (RTI) 0100_{16}

0	1	2	3	4	5	6	7	8	9–15
0	0	0	0	0	0	0	1	0	ctl

Operation: Set INTEN (interrupt enable flag) (PC) ← (STK) + ctl.

Description: The interrupt enable flag (INTEN) is set. The contents of PC are replaced by the sum of ctl and the contents of STK. Program control is transferred to the location specified by the new contents of PC. (RTI is used primarily to exit from an interrupt routine.)

RETURN FROM SUBROUTINE (RTS) 0200_{16}

0	1	2	3	4	5	6	7	8	9–15
0	0	0	0	0	0	1	0	0	ctl

Operation: (PC) ← (STK) + ctl

Description: The contents of PC are replaced by the sum of ctl and the contents of STK. Program control is transferred to the location specified by the new contents of PC. (RTS is used primarily to return from subroutines entered by JSR.)

NOTE

For both the preceding instructions (RTI and RTS), the ctl value is an unsigned 7-bit number.

2-22. SHIFT INSTRUCTIONS.

Four instructions comprise this group. All four instructions may be used with the Link (L) bit by setting the SEL flag. This is accomplished with a Set Flag (SFLG) instruction before executing the shift or rotate instruction. Examples of shifting with and without SEL set are given in diagram form for each instruction in the descriptions that follow. Note that the SEL flag also affects the BOC instructions as indicated in Table 2-8.

The shift instructions are summarized in Table 2-9, and th word format is shown in Figure 2-8.

All shift and rotate operations may be carried out with any of the four general-purpose accumulators, AC0, 1, 2, or 3.

Programming Note: Since the Shift instructions and the BCYOF instructions behave differently according to the value of SEL, it is important to establish the required setting. The System II Operating System considers the normal usage to be SEL = 0 and resets SEL on return from all user service calls.

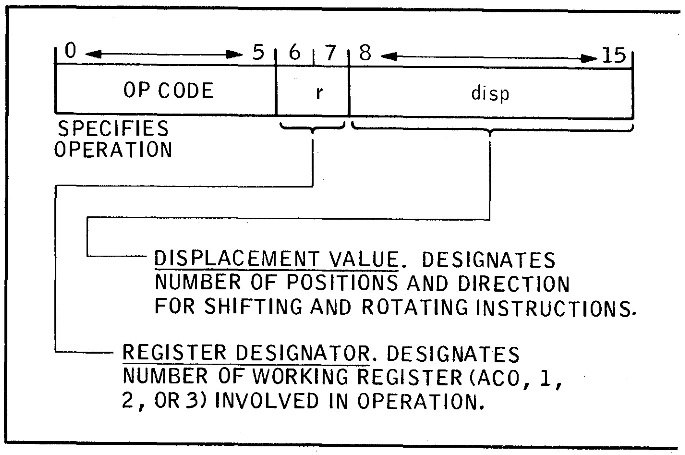

0
5
6
7
8
15
OP CODE
r
disp
SPECIFIES OPERATION
DISPLACEMENT VALUE. DESIGNATES NUMBER OF POSITIONS AND DIRECTION FOR SHIFTING AND ROTATING INSTRUCTIONS.
REGISTER DESIGNATOR. DESIGNATES NUMBER OF WORKING REGISTER (AC0, 1, 2, OR 3) INVOLVED IN OPERATION.

Figure 2-8. Shift Instruction Format

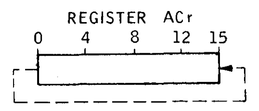
ROTATE LEFT (ROL) (for disp > 0) 5800_{16}

0 – 5	6 – 7	8 – 15
0 1 0 1 1 0	r	disp

SEL = 0

Operation: $(ACr_{15}) \leftarrow (ACr_0),\ (ACr_n) \leftarrow (ACr_{n+1})$

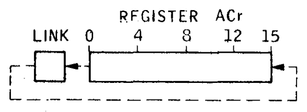
Description: The contents of ACr are shifted around to the left disp times. (ACr_0) replaces (ACr_{15}) for each shift.



SEL = 1

Operation: $(L) \leftarrow (ACr_0),\ (ACr_{15}) \leftarrow (L),\ (ACr_n) \leftarrow (ACr_{n+1})$

Description: The contents of ACr are shifted around to the left disp times. (L) replaces (ACr_{15}), and (ACr_0) replaces (L) for each shift.



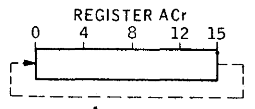
ROTATE RIGHT (ROR) (for disp < 0) 5800_{16}

0 – 5	6 – 7	8 – 15
0 1 0 1 1 0	r	disp

SEL = 0

Operation: $(ACR_0) \leftarrow (ACr_{15}),\ (ACr_n) \leftarrow (ACr_{n-1})$

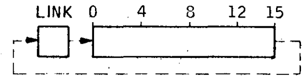
Description: The contents of ACr are shifted around to the right disp times. (ACr_{15}) replaces (ACr_0) for each shift.



SEL = 1

Operation: $(ACr_0) \leftarrow (L),\ (L) \leftarrow (ACr_{15})$
$(ACr_n) \leftarrow (ACr_{n-1})$

Description: The contents of ACr are shifted around to the right disp times. (L) replaces (ACr_0), and (ACr_{15}) replaces (L) for each shift.



NOTE

There is only one rotate operation code. The direction of the rotate is implied by the sign of the 8-bit displacement field. However, the assembler recognizes two distinct mneumonics (ROL, ROR) and expects a positive shift count for either.

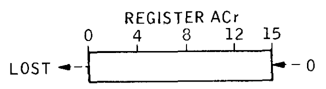
SHIFT LEFT (SHL) (for disp > 0) $5C00_{16}$

0 – 5	6 – 7	8 – 15
0 1 0 1 1 1	r	disp

SEL = 0

Operation: $(ACr_n) \leftarrow (ACr_{n+1}),(ACr_{15}) \leftarrow 0$

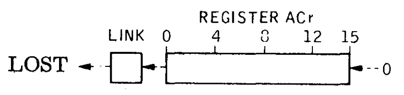
Description: The contents of ACr are shifted to the left disp times. (ACr_0) is lost, and zero replaces (ACr_{15}) for each shift.



SEL = 1

Operation: $(L) \leftarrow (ACr_0),(ACr_n) \leftarrow (ACr_{n+1}),$
$(ACr_{15}) \leftarrow 0$

Description: The contents of ACr are shifted to the left disp times. (L) is lost. (ACr_0) replaces (L), and zero replaces (ACr_{15}) for each shift.



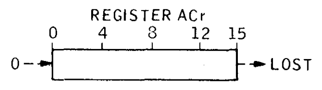
SHIFT RIGHT (SHR) (for disp < 0) $5C00_{16}$

0 – 5	6 – 7	8 – 15
0 1 0 1 1 1	r	disp

SEL = 0

Operation: $(ACr_0) \leftarrow 0,\ (ACr_n) \leftarrow (ACr_{n-1})$

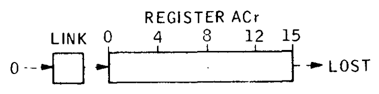
Description: The contents of ACr are shifted to the right disp times. (ACr_{15}) is lost, and zero replaces (ACr_0) for each shift.



SEL = 1

Operation: $(L) \leftarrow 0,\ (ACr_0) \leftarrow (L),\ (ACr_n) \leftarrow (ACr_{n-1})$

Description: The contents of ACr are shifted to the right disp times. (ACr_{15}) is lost, (L) replaces (ACr_0), and zero replaces (L) for each shift.



NOTE

There is only one shift operation code. The direction of the shift is implied by the sign of the 8-bit displacement field. However, the assembler recognizes two distinct mneumonics (SHL, SHR) and expects a positive shift count for either.

Table 2-9. Shift Instructions

INSTRUCTION	OPERATION CODE	OPERATION		ASSEMBLER FORMAT
		SEL = 0	SEL = 1	
ROTATE LEFT (disp > 0)	010110	$(ACr_{15}) \leftarrow (ACr_0)$, $(ACr_n) \leftarrow (ACr_{n+1})$	$(L) \leftarrow (ACr_0)$ $(ACr_{15}) \leftarrow (L)$ $(ACr_n) \leftarrow (ACr_{n+1})$	ROL r, m
ROTATE RIGHT (disp < 0)	010110	$(ACr_0) \leftarrow (ACr_{15})$, $(ACr_n) \leftarrow (ACr_{n-1})$	$(ACr_0) \leftarrow (L)$. $(L) \leftarrow (ACr_{15})$, $(ACr_n) \leftarrow (ACr_{n-1})$	ROR r, m
SHIFT LEFT (disp > 0)	010111	$(ACr_n) \leftarrow (ACr_{n+1})$, $(ACr_{15}) \leftarrow 0$	$(L) \leftarrow (ACr_0)$, $(ACr_n) \leftarrow (ACr_{n+1})$, $(ACr_{15}) \leftarrow 0$	SHL r, m
SHIFT RIGHT (disp < 0)	010111	$(ACr_0) \leftarrow 0$, $(ACr_n) \leftarrow (ACr_{n-1})$	$(L) \leftarrow 0$, $(ACr_0) \leftarrow (L)$, $(ACr_n) \leftarrow (ACr_{n-1})$	SHR r, m

NOTE

For all shift and rotate instructions, "m" denotes the number of positions to be shifted or rotated, and is equal to the absolute value of disp. See example 3 in Appendix B.

Table 2-10. Register Instructions

INSTRUCTION	OPERATION CODE	OPERATION	ASSEMBLER FORMAT
Register and Stack			
PUSH ONTO STACK	010000	$(STK) \leftarrow (ACr)$	PUSH r
PULL FROM STACK	010001	$(ACr) \leftarrow (STK)$	PULL r
EXCHANGE REGISTER AND STACK	010101	$(STK) \leftarrow (ACr)$, $(ACr) \leftarrow (STK)$	XCHRS r
Register and Immediate			
LOAD IMMEDIATE	010011	$(ACr) \leftarrow$ disp (sign extended)	LI r, disp
ADD IMMEDIATE, SKIP IF ZERO	010010	$(ACr) \leftarrow (ACr) +$ disp (sign extended), OV, CY: if $(ACr) = 0$, $(PC) \leftarrow (PC)+1$	AISZ r, disp
COMPLEMENT AND ADD IMMEDIATE	010100	$(ACr) \leftarrow \sim(ACr) +$ disp (sign extended)	CAI r, disp

INSTRUCTION	OPERATION CODE			OPERATION	ASSEMBLER FORMAT
	OP1	OP2	OP3		
Register to Register	0-3	8	13-15		
REGISTER ADD	0011	0	000	$(ACdr) \leftarrow (ACdr) + (ACsr)$, OV, CY	RADD sr, dr
REGISTER EXCHANGE	0011	1	000	$(ACsr) \leftarrow (ACdr)$, $(ACdr) \leftarrow (ACsr)$	RXCH sr, dr
REGISTER COPY	0011	1	001	$(ACdr) \leftarrow (ACsr)$	RCPY sr, dr
REGISTER EXCLUSIVE OR	0011	1	010	$(ACdr) \leftarrow (ACsr) \veebar (ACdr)$	RXOR sr, dr
REGISTER AND	0011	1	011	$(ACdr) \leftarrow (ACsr) \wedge (ACdr)$	RAND sr, dr
COMPUTE BLOCK CHECK CHARACTER	0011	1	100	$(ACdr) \leftarrow$ BCC (ACsr, ACdr)	RBCC8 sr, dr

2-23. REGISTER INSTRUCTIONS.

The eleven instructions in this group are summarized in Table 2-10. The three word formats required are shown in Figure 2-9.

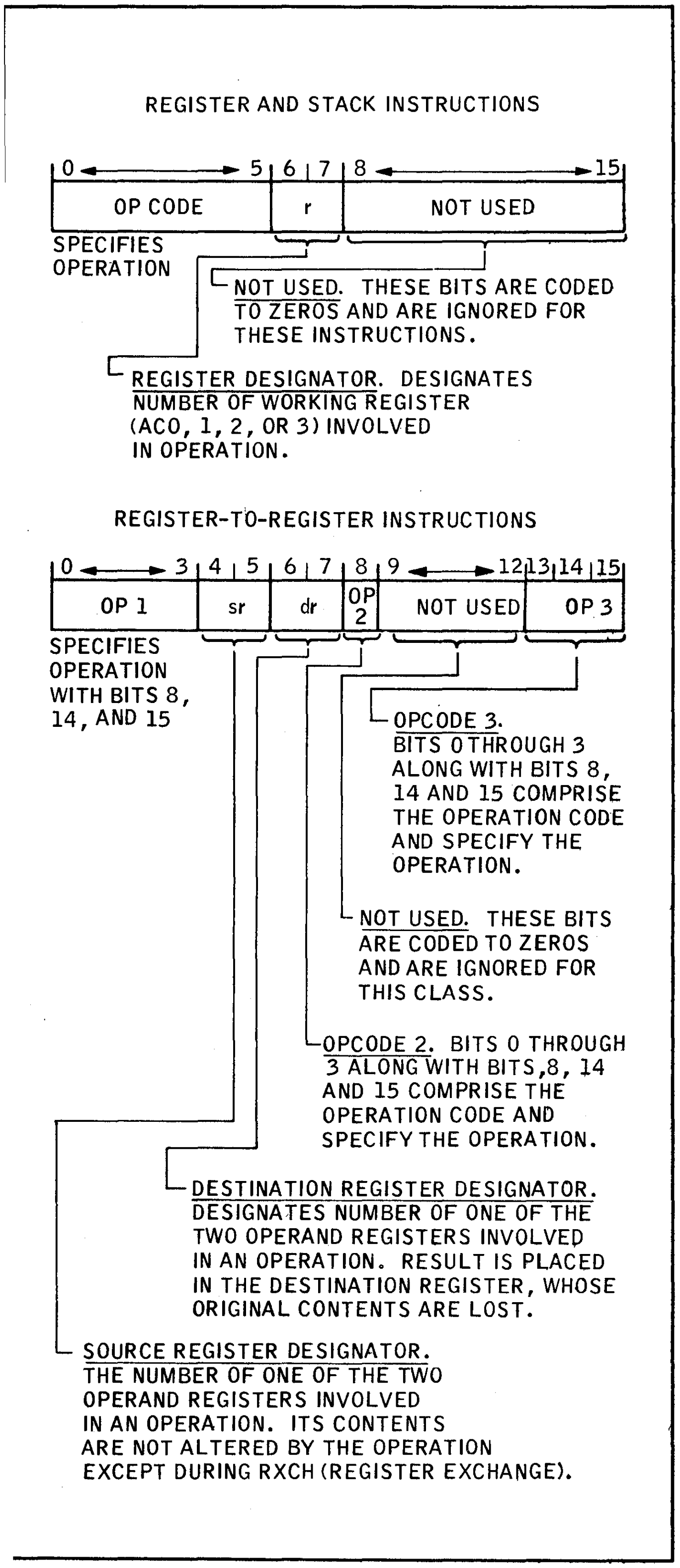

REGISTER AND STACK INSTRUCTIONS
0 5 6 7 8 15
OP CODE
r
NOT USED
SPECIFIES OPERATION
NOT USED. THESE BITS ARE CODED TO ZEROS AND ARE IGNORED FOR THESE INSTRUCTIONS.
REGISTER DESIGNATOR. DESIGNATES NUMBER OF WORKING REGISTER (AC0, 1, 2, OR 3) INVOLVED IN OPERATION.
REGISTER-TO-REGISTER INSTRUCTIONS
0 3 4 5 6 7 8 9 12 13 14 15
OP 1
sr
dr
OP 2
NOT USED
OP 3
SPECIFIES OPERATION WITH BITS 8, 14, AND 15
OPCODE 3. BITS 0 THROUGH 3 ALONG WITH BITS 8, 14 AND 15 COMPRISE THE OPERATION CODE AND SPECIFY THE OPERATION.
NOT USED. THESE BITS ARE CODED TO ZEROS AND ARE IGNORED FOR THIS CLASS.
OPCODE 2. BITS 0 THROUGH 3 ALONG WITH BITS ,8, 14 AND 15 COMPRISE THE OPERATION CODE AND SPECIFY THE OPERATION.
DESTINATION REGISTER DESIGNATOR. DESIGNATES NUMBER OF ONE OF THE TWO OPERAND REGISTERS INVOLVED IN AN OPERATION. RESULT IS PLACED IN THE DESTINATION REGISTER, WHOSE ORIGINAL CONTENTS ARE LOST.
SOURCE REGISTER DESIGNATOR. THE NUMBER OF ONE OF THE TWO OPERAND REGISTERS INVOLVED IN AN OPERATION. ITS CONTENTS ARE NOT ALTERED BY THE OPERATION EXCEPT DURING RXCH (REGISTER EXCHANGE).

Figure 2-9. Register Instruction Formats

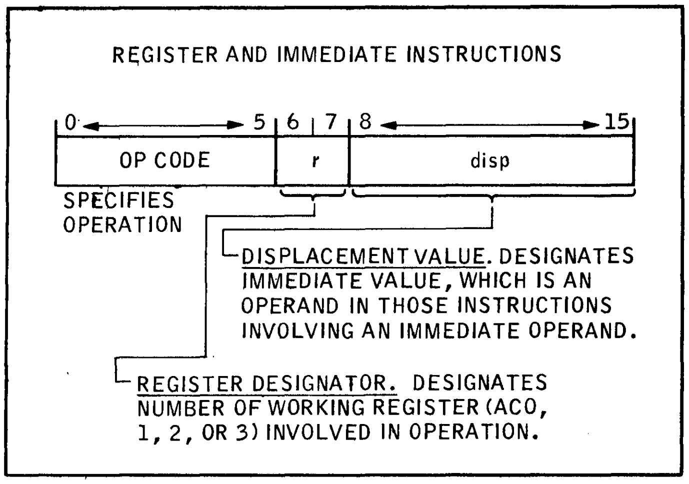

REGISTER AND IMMEDIATE INSTRUCTIONS
0 5 6 7 8 15
OP CODE
r
disp
SPECIFIES OPERATION
DISPLACEMENT VALUE. DESIGNATES IMMEDIATE VALUE, WHICH IS AN OPERAND IN THOSE INSTRUCTIONS INVOLVING AN IMMEDIATE OPERAND.
REGISTER DESIGNATOR. DESIGNATES NUMBER OF WORKING REGISTER (AC0, 1, 2, OR 3) INVOLVED IN OPERATION.

Figure 2-9. Register Instruction Formats (Continued)

PUSH ONTO STACK (PUSH) 4000_{16}

0					5	6 7	8 … 15
0	1	0	0	0	0	r	NOT USED

Operation:	(STK)◄—(ACr)
Description:	The stack is pushed by the contents of the register (AC0, 1, 2, or 3) designated by r. Thus, the top of the stack then holds the contents of ACr, and the contents of all other levels in the stack are moved down one level. If the stack is full before the push occurs, the contents of the lowest level are lost. The initial contents of ACr are unaltered.

Programming Note: See programming note for Jump to Subroutine instruction for caution on use of PUSH instruction.

PULL FROM STACK (PULL) 4400_{16}

0					5	6 7	8 … 15
0	1	0	0	0	1	r	NOT USED

Operation:	(ACr)◄—(STK)
Description:	The stack is pulled. The contents from the top of the stack replace the contents of register number r (AC0, 1, 2, or 3). The initial contents of ACr are lost. The contents of each level of the stack moves up one level. Zeros enter the bottom of the stack.

EXCHANGE REGISTER AND STACK (XCHRS) 5400_{16}

0					5	6 7	8 … 15
0	1	0	1	0	1	r	NOT USED

Operation:	(STK)◄—(ACr), (ACr)◄—(STK)
Description:	The contents of the top of the stack STK and the register designated by r (AC0, 1, 2, or 3) are exchanged.

LOAD IMMEDIATE (LI) $4C00_{16}$

0 5	6 7	8 15
0 1 0 0 1 1	r	disp

Operation: (ACr)$\leftarrow$ disp (sign extended)

Description: The value of disp with sign bit 8 extended through bit 0 replaces the contents of ACr (AC0, 1,2, or 3). The initial contents of ACr are lost. The immediate operand range is -128 to +127.

ADD IMMEDIATE, SKIP IF ZERO (AISZ) 4800_{16}

0 5	6 7	8 15
0 1 0 0 1 0	r	disp

Operation: (ACr)$\leftarrow$(ACr) + disp (sign extended), OV, CY. If new (ACr) = 0, (PC)$\leftarrow$(PC) + 1

Description: The contents of register ACr are replaced by the sum of the contents of ACr and disp (sign bit 8 extended through bit 0). The Initial contents of ACr are lost. The overflow and carry flags are set according to the result of the operation. If the new contents of ACr equal zero, the contents of PC are incremented by 1, thus skipping the next memory location. The immediate operand range is -128 to +127. Refer to para. 2-8.

Programming note: This instruction is useful for sensing zero/non-zero in AC1-AC3. (Code with a zero displacement field.)

COMPLEMENT AND ADD IMMEDIATE (CAI) 5000_{16}

0 5	6 7	8 15
0 1 0 1 0 0	r	disp

Operation: (ACr)$\leftarrow$ ~ (ACr) + disp (sign extended)

Description: The contents of register ACr are one's complemented and then added to disp (sign bit 8 extended through bit 0). The result is then stored in ACr. The initial contents of ACr are lost. The immediate operand range is -128 to +127. Note that the carry and overflow flags are not affected by this instruction.

Programming Note: CAI with an immediate operand of 1 provides the two's complement of ACr.

REGISTER ADD (RADD) 3000_{16}

0 3	4 5	6 7	8	9	13 14 15
0 0 1 1	sr	dr	0	NOT USED	0 0 0

Operation: (ACdr)$\leftarrow$ (ACsr) + (ACdr), OV, CY

Description: The contents of the destination register ACdr (AC0, 1, 2, or 3) are replaced by the sum of the contents of ACdr and the source register ACsr (AC0, 1, 2, or 3). The initial contents of ACdr are lost, and the contents of ACsr are unaltered. The overflow and carry flags are set according to the result of the operation. Refer to para. 2-8.

REGISTER EXCLUSIVE OR (RXOR) 3082_{16}

0 3	4 5	6 7	8	9	13 14 15
0 0 1 1	sr	dr	1	NOT USED	0 1 0

Operation: (ACdr) $\leftarrow$(ACdr)$\veebar$(ACsr)

Description: The contents of the destination register ACdr (AC0, 1, 2, or 3) are replaced by the result of exclusively ORing the contents of ACdr with the contents of the source register ACsr (AC0, 1, 2, or 3). The initial contents of ACsr are unaltered.

REGISTER AND (RAND) 3083_{16}

0 3	4 5	6 7	8	9	13 14 15
0 0 1 1	sr	dr	1	NOT USED	0 1 1

Operation: (ACdr)$\leftarrow$ (ACdr) $\wedge$ (ACsr)

Description: The contents of the destination register ACdr (AC0, 1, 2, or 3) are replaced by the result of ANDing the contents of ACdr with the contents of the source register ACsr (AC0, 1, 2, or 3). The initial contents of ACdr are lost, and the initial contents of ACsr are unaltered.

REGISTER EXCHANGE (RXCH) 3080_{16}

0 3	4 5	6 7	8	9	13 14 15
0 0 1 1	sr	dr	1	NOT USED	0 0 0

Operation: (ACsr)$\leftarrow$(ACdr), (ACdr)$\leftarrow$(ACsr)

Description: The contents of ACsr (AC0, 1, 2, or 3) and ACdr (AC0, 1, 2, or 3) are exchanged.

REGISTER COPY (RCPY) 3081_{16}

0 3	4 5	6 7	8	9	13 14 15
0 0 1 1	sr	dr	1	NOT USED	0 0 1

Operation: (ACdr)$\leftarrow$(ACsr)

Description: The contents of the destination register ACdr (AC0, 1, 2, or 3) are replaced by the contents of the source register ACsr (AC0, 1, 2, or 3). The initial contents of ACdr are lost, and the initial contents of ACsr are unaltered.

COMPUTE BLOCK CHECK CHARACTER (RBCC8) 3084_{16}

0 1 2 3	4 5	6 7	8	9 10 11 12	13 14 15
0 0 1 1	sr	dr	1	NOT USED	1 0 0

Operation: (ACdr) $\leftarrow$ BCC (ACsr, ACdr)

Description: The RBCC8 instruction is used to compute block checksum characters for communication messages. It is a register to register instruction that operates as shown above.

The source register (specified by the sr field) is the polynomial coefficients represented in binary with the bits reversed from right to left. Thus, bit-0 of the reg is the low order bit of the polynomial, and bit-15 of the reg is the next to the high order bit of the polynomial. The highest order term of the polynomial is always discarded. Thus, the CRCC16 polynomial used in binary synchronous communications ($X^{16}+X^{15}+X^{2}+1$) is represented in binary as 1010000000000001.

The destination register (specified by the dr field) is the partial BCC which will be updated by this instruction. Before executing the instruction, the new character to be included into the BCC must have been exclusive-ORed onto bits 8-15 of the partial BCC.

The instruction procedure consists of 8 iterations of the following: Shift the partial BCC (in the dr register) right 1 bit; if the bit shifted off is a 1, exclusive-OR the polynomial (in the sr register) onto the shifted partial BCC (in the dr register). When the instruction is complete, the dd register will contain the new BCC which consists of two check characters, the first in bits 8-15, and the second in bits 0-7.

The instruction can be used for computing the BCC on two characters at a time. In this mode, it is necessary to exclusive-OR the pair of new characters (the first in bits 8-15 and the second in bits 0-7) onto the partial BCC and then execute the RBCC8 instruction twice.

2-24. INPUT/OUTPUT INSTRUCTIONS.

The input/output instructions described in this section provide the facilities for programmed data transfer in the J500 machine environment. The instructions in this group are summarized in Table 2-11. This method of I/O involves the transfer of 16-bits of data between a peripheral device and accumulator AC0. The specific device involved is determined by addressing information in accumulator AC3. Note specifically that programmed data transfer involves two fixed register assignments: AC0 for data and AC3 for addressing control information.

Table 2-11. Input/Output Instructions

ASSEMBLER FORMATS	INSTRUCTION VALUE	OPERATION
DIA	0401_{16}	(AR) $\leftarrow$ (AC3) (AC0) $\leftarrow$ (IOREGA)
DIB	0402_{16}	(AR) $\leftarrow$ (AC3) (AC0) $\leftarrow$ (IOREGB)
DIC	0404_{16}	(AR) $\leftarrow$ (AC3) (AC0) $\leftarrow$ (IOREGC)
DOA	0601_{16}	(AR) $\leftarrow$ ctl + ((AC0) $\rightarrow$ (IOREGA)
DOB	0602_{16}	(AR) $\leftarrow$ (AC3) (AC0) $\rightarrow$ (IOREGB)
DOC	0604_{16}	(AR) $\leftarrow$ (AC3) (AC0) $\rightarrow$ (IOREGC)
RDDNE	0620_{16}	CPU $\leftarrow$ Done flag
RDBSY	0640_{16}	CPU $\leftarrow$ Busy flag
INTA	0410_{16}	(AC0) $\leftarrow$ Device Add of highest priority device that generated interrupt
IORST	0420_{16}	Reset to all devices
MSKO	0608_{16}	(AC0) All devices to disable interrupt masks
NIO	0600_{16}	No data transfer

For a complete understanding of the action of the instructions in this section, the interface specification for the devices of interest must be consulted. The design of a specific device controller determines the significance of the various I/O instructions as they relate to that device. Interface specifications for the CRT/Keyboard controller, Universal Disk Drive controller, Communications controller, Auto Dial controller and Printer controller are contained in other sections of this manual.

The general conventions that apply to all J500 controllers are presented here. Some controllers provide both a Busy and a Done flag which reflect the state of the controller and its device(s). The Busy flag when true indicates those intervals in time when an operation is being performed by the controller and it is not able to accept any other. Similarly, the Done flag when true indicates that a previous operation was completed and the controller is available for the next operation. Consult the device specification of interest for a more detailed explanation of the Busy and Done flags. These flags may be interrogated with the RDBSY and RDDNE instructions.

At the completion of an operation, the J500 device controllers will attempt to cause an interrupt. This interrupt consists of a "jump and store" to memory location one after completion of the current instruction. The act of jumping to location one disables further interrupts from occurring until interrupts are enabled by a RTI or INTEN instruction. The state of the Interrupt Enable Flag and Mask Out Register are interrogated before each interrupt. If the Interrupt Enable Flag is OFF, no interrupts can be generated. If the Interrupt Enable Flag is ON, each device controller will furthermore interrogate the Mask Out Register before causing an interrupt. If the specific bit in the Mask Out Register corresponding to the device controller which is attempting the interrupt is ON, that device will not generate an interrupt. Interrupts from an "interrupt disabled" controller will be held suspended (not lost) until that controller is enabled via another MSKO or INTEN instruction.

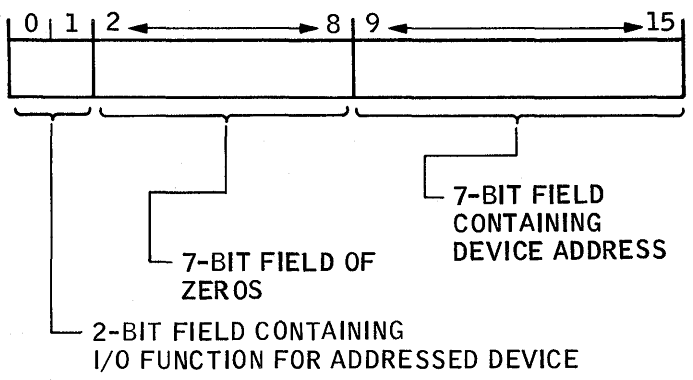
An INTA instruction will cause the interrupting device to place its 7 bit Device Code in AC0. In the event that two devices have interrupts pending, the highest priority device will transmit its device code. The priority of the devices is determined by the hardware and the priorities are listed in Table 2-12.

Table 2-12. Interrupt Priorities

Priority	Interrupt
1 (Highest)	Communications Line 1
2	Communications Line 2
3	Keyboard
4	Auto Dial
5	Character Printer
6	Cartridge Drives
7	Floppy Drives
8	Real Time Clock
9 (Lowest)	Line Printer

If none of the devices in Table 2-12 is reporting an interrupt and the stack is full, a device code of Hex 00 will be returned to AC0 on the completion of the INTA instruction.

Device controllers are uniquely identified by means of a 7-bit "device address." Zero is not permitted, leaving 127 possible device addresses in the J500 I/O architecture. The device addresses assigned to the J500 are listed in Table 2-13. When using an I/O instruction which addresses a specific device controller, accumulator AC3 must be set up in the following fashion:


0 1 2 8 9 15
7-BIT FIELD CONTAINING DEVICE ADDRESS
7-BIT FIELD OF ZEROS
2-BIT FIELD CONTAINING I/O FUNCTION FOR ADDRESSED DEVICE

In addition to the device control provided by the I/O instructions themselves, certain I/O functions may be generated simultaneous with any I/O instruction by simply encoding the 2-bit function field in AC3. The I/O functions permitted are:

00 -	no function
01 -	start
10 -	clear
11 -	no function

For the purpose of the Start and Clear functions for each controller, refer to the individual controller descriptions in this manual.

Table 2-13. Device Addresses

Device	Device Code (Hex)
Real Time Clock	0C
Character Printer	0E
Line Printer	0F
Floppy Drive No. 1	14
Floppy Drive No. 2	15
Floppy Drive No. 3	16
Floppy Drive No. 4	17
Cartridge Drive No. 1	18
Cartridge Drive No. 2	19
Cartridge Drive No. 3	1A
Cartridge Drive No. 4	1B
Auto Dial	1D
Communications Line No. 1	1E
Communications Line No. 2	1F
Keyboard	28

DATA IN A (DIA) 0401_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 0 0	0 0 0 0	0 0 0 1

Operation:	(AR) ← (AC3), (AC0) ← (IOREGA)
Description:	A particular device controller is addressed by (AC3). The peripheral device controller responds by transferring the contents of its A-register to the processor AC0 register.
	Any function specified in AC3 is output to the addressed device controller.

DATA IN B (DIB) 0402_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 0 0	0 0 0 0	0 0 1 0

Operation:	(AR) ← (AC3), (AC0) ← (IOREGB)
Description:	A particular device controller is addressed by (AC3). The peripheral device controller responds by transferring the contents of its B-register to the processor AC0 register.
	Any function specified in AC3 is output to the addressed device controller.

DATA IN C (DIC) 0404_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 0 0	0 0 0 0	0 1 0 0

Operation: (AR)◄—(AC3), (AC0) ◄— (IOREGC)

Description: A particular device controller is addressed by AC3. The peripheral device controller responds by transferring the contents of its C-register to the processor AC0 register.

Any function specified in AC3 is output to the addressed device controller.

DATA OUT A (DOA) 0601_{16}

0 3	4	8 11	12 15
0 0 0 0	0 1 1 0	0 0 0 0	0 0 0 1

Operation: (AR) ◄— (AC3), (AC0) —► (IOREGA)

Description: A particular device controller is addressed by (AC3). The peripheral device controller responds by transferring the contents of the processor AC0 register to its A-register.

Any function specified in AC3 is output to the addressed device controller.

DATA OUT B (DOB) 0602_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 1 0	0 0 0 0	0 0 1 0

Operation: (AR)◄—(AC0) —► (IOREGB)

Description: A particular device controller is addressed by (AC3). The peripheral device controller responds by transferring the contents of the processor AC0 register to its B-register.

Any function specified in AC3 is output to the addressed device controller.

DATA OUT C (DOC) 0604_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 1 0	0 0 0 0	0 1 0 0

Operation: (AR)◄—(AC3), (AC0) —► (IOREGC)

Description: A particular device controller is addressed by (AC3). The peripheral device controller responds by transferring the contents of the processor AC0 register to its C-register.

Any function specified in AC3 is output to the addressed device controller.

READ DONE FLAG (RDDNE) 0620_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 1 0	0 0 1 0	0 0 0 0

Operation: (ACr)◄—(AC3) CPU Condition Code 13◄—Done Flag

Description: A particular device controller is addressed by (AC3). The peripheral device controller responds by transferring the state of its Done flag into the CPU where it may be tested with a BIODS instruction (BOC 13).

READ BUSY FLAG (RDBSY) 0640_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 1 0	0 1 0 0	0 0 0 0

Operation: (AR)◄—(AC3) CPU Condition Code 12◄—Busy Flag

Description: A particular device controller is addressed by (AC3). The peripheral device controller responds by transferring the state of its Busy flag into the CPU where it may be tested with a BIOBS instruction (BOC 12).

INTERRUPT ACKNOWLEDGE (INTA) 0410_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 0 0	0 0 0 1	0 0 0 0

Operation: AC0 ◄—highest priority device address of those devices requesting interrupt.

Description: This instruction does not address a particular device controller. The highest priority device which is requesting an interrupt will transfer its 7-bit device address into the low order (bits 10-15) of processor AC0 register. This instruction has no effect on the state of any device controller.

NOTE

Refer to the section describing Stack Full Sensing as the INTA instruction does affect this CPU feature.

I/O RESET (IORST) 0420_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 0 0	0 0 1 0	0 0 0 0

Operation: Reset all device controllers.

Description: This is a broadcast type instruction that affects all controllers attached to the J500. It is equivalent to a CLEAR function on all devices.

MASK OUT (MSKO) 0608_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 1 0	0 0 0 0	1 0 0 0

Operation: (AC0) → I/O bus

Description: This instruction is a broadcast type since it applies simultaneously to all controllers attached to the J500. The contents of AC0 is saved in the MSKO register where each device controller samples a single bit of the 16-bit register. This bit is used to set the state of the controllers interrupt disable mechanism. When the sampled bit is one, interrupts will be disabled for that controller. Conversely, when zero interrupts will be enabled from the controller. More than one controller may sample the same bit on the bus. This mechanism can be used to implement a priority interrupt handling scheme. See Table 2-13 for mask bit assignments.

NOTE

There is no Mask In instruction; if the current status needs to be known it must be kept track of in some other manner. (e.g. store in memory)

Table 2-14. Mask Out Bits

Bit	Device
8	Comm Controller
8	Auto Dial
14	Keyboard
12	Character Printer
12	Line Printer
7	Cartridge Drives
7	Floppy Drives
13	Real Time Clock

NO INPUT/OUTPUT (NIO) 0600_{16}

0 3	4 7	8 11	12 15
0 0 0 0	0 1 1 0	0 0 0 0	0 0 0 0

Operation: (AR) ← (AC3)

Description: A particular device controller is addressed by (AC3). No data transfer occurs between the processor AC0 register and the device controller. This instruction allows I/O functions to be issued to a controller without any data transfer.

2-25. HALT AND FLAG INSTRUCTIONS.

The J500 processor architecture includes a register of status flags. This 16-bit register contains the link, overflow, and carry flags as well as 13 other flag bits which currently are not used. Figure 2-10 and Table 2-15 illustrate the layout of these flags in the register. Two instructions are provided to allow the saving, restoring, and direct manipulation of these flags. Note that the SELECT flag is not included in this status flag register.

Various control flags are also included in the J500 processor's architecture. Table 2-16 itemizes these flags; two are of primary importance. Flag 001 is the INTEN flag which controls the processors interrupt recognition (see INTERRUPTS) while flag 002 is the SELECT flag which affects shifts/rotates and condition code selection for the BCYOF (BOC) instruction. Note that these control flags are not available in register form like the Status flags, but rather each must be manipulated directly with SFLG and PFLG instructions. SFLG is used to turn on a control flag while PFLG will leave the specified control flag off.

Halt instructions and illegal instructions are handled in a fashion similar to interrupts. When one of these instructions is executed, "a jump and store" to memory location 2 is performed and the Interrupt Enable flag (INTEN) is cleared. This allows the operating system to trap these illegal operations and not cause the machine to "hang up".

NOTE

If a Halt instruction is desired, memory location 2 should be used for the Halt instruction.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	OV	CY	GF	GF	GF	GF	GF	GF	GF	GF	GF	GF	GF	GF	GF

Figure 2-10. Configuration of Status Flags

PULL STATUS FLAGS FROM STACK (PULLF) 0280_{16}

0 8	9 15
0 0 0 0 0 0 1 0 1	NOT USED

Operation: (STATUS FLAGS) ← (STK)

Description: The contents of the status flags are replaced by the contents of the top of the stack (STK). See Figure 2-10 for the configuration of processor flags on the stack and Table 2-15 for processor-flag definitions. The previous contents of lower levels of the stack are pulled up by one level with zeros replacing the contents of the lowest level.

Table 2-15. Halt and Flag Instructions

OPERATION CODE		OPERATION	ASSEMBLER FORMATS
000000000		Processor halts	HALT
000000001		(STK)←(STATUS FLAGS)	PUSHF
000000101		(STATUS FLAGS)→(STK)	PULLF
OPERATION CODE		OPERATION	ASSEMBLER FORMAT
OP1	OP2		
00001	0	fc set; (AR)←ctl	SFLG fc
00001	1	fc pulsed; (AR)←ctl	PFLG fc

Table 2-16. Status Flags

BIT POSITION	FLAG NAME	MNEMONIC	SIGNIFICANCE
0	Link	L	Used for double-word shifts
1	Overflow	OV	Set if an arithmetic overflow occurs.
2	Carry	CY	Set if a carry occurs (from most significant bit) during an arithmetic operation.
3 through 15	General-Purpose Flags	GF	Use specified by programmer.

Table 2-17. Control Flag Codes

FC	FLAG MNEMONIC	SIGNIFICANCE
000	F0	Not used
001	INT EN	Interrupt Enable (enables/disables external interrupt recognition)
010	SEL	Select (affects shifts, rotates, and BCYOF)
011	F3	Not used
100	F4	Not used
101	F5	Not used.
110	F6	Not used
111	F7	Not used
NOTE: The flag designated by the flag code (fc) is set or pulsed.		

SET FLAG (SFLG) 0800_{16}

0–4	5–7	8	9–15
0 0 0 0 1	fc	0	NOT USED

Operation: fc set

Description: The control flag designated by the flag code fc is set. Flag codes are defined in Table 2-16.

PULSE FLAG (PFLG) 0880_{16}

0–4	5–7	8	9–15
0 0 0 0 1	fc	1	NOT USED

Operation: fc cleared

Description: The control flag designated by the flag code fc is cleared. Flag codes are defined in Table 2-16.

Programming Note: This instruction is used primarily to reset flags set by the SFLG instruction.

PUSH STATUS FLAGS ONTO STACK (PUSHF) 0080_{16}

0–8	9–15
0 0 0 0 0 0 0 0 1	NOT USED

Operation: (STK)←(STATUS FLAGS)

Description: The contents of the top of the stack (STK) are replaced by the contents of the status flags. See Figure 2-10 for the configuration of processor flags on the stack and Table 2-15 for processor flag definitions. The previous contents of the top of the stack and lower levels are pushed down one level. The contents of the lowest level of the stack are lost.

HALT (HALT) 0000_{16}

0–8	9–15
ALL ZEROS	NOT USED

Operation: (STK) ← (PC) (PC) ← 0002_{16}

Description: The content of PC are stored on top of the stack. The PC is replaced with Hex 0002. The Interrupt Enable flag (INTEN) is cleared.

NOTE

(1) SFLG and PFLG refer to control flags. These flags should not be confused with the STATUS flags, referenced by PUSHF and PULLF.

(2) Since the INTEN (interrupt enable) appears as a control flag (F1, Table 2-16), a SFLG may be used as an interrupt enable (INTEN) instructions and a PFLG may be used as an interrupt disable (INTDS).

2-26. INTERRUPT SYSTEM.

The J500 interrupt facilities allow the processor to be directed from an arbitrary sequence of code to a specific instruction sequence in response to an external asynchronously occurring event. A machine stack full condition, which is an internal event, can also be handled through the interrupt facilities.

These interrupt facilities include the means to enable or disable at the processor's discretion, the recognition of any interrupt requests. This feature is called "interrupt enable" and is manipulated with two instructions:

INTERRUPT ENABLE (INTEN) 0900_{16}

0 3	4 7	8 11	12 15
0 0 0 0	1 0 0 1	0 0 0 0	0 0 0 0

Operation: Set Flag 001_2

Description: To enable the processor's interrupt recognition mechanism immediately following this instruction. If interrupts are already enabled, this instruction has no effect.

INTERRUPT DISABLE (INTDS) 0980_{16}

0 3	4 7	8 11	12 15
0 0 0 0	1 0 0 1	1 0 0 0	0 0 0 0

Operation: Clear Flag 001_2

Description: To immediately disable the processor's interrupt recognition mechanism. If interrupts are already disabled this information has no effect.

Between every instruction the CPU samples the interrupt request signal. If an interrupt request is present along with interrupt enable (INT EN true), the processor will automatically disable interrupts (INTDS), push the current instruction address (PC) into the machine stack, and transfer CPU control to location 1 in main memory. This memory location is dedicated to interrupt handling. If this facility is used, this location must contain linkage to an interrupt service routine.

In general an interrupt service routine must do certain basic things:

1. Save any part of the machine state that might be destroyed by the interrupt servicing.
2. Identify the source of the interrupt request by either polling or use of the INTA instruction.
3. Provide service required and clear the interrupt request from the specific source. An interrupt from a device controller may be cleared by issuing either a clear I/O function or by initiating another controller operation with a Start I/O function, either of which will cause the Done flag to be reset. I/O reset will also clear the interrupt request.
4. Restore any saved machine state and return with an RTI instruction. Note that this is not equivalent to an INTEN and RTS due to the timing of the interrupt enable relative to the stack pop $\rightarrow$ PC operation.

2-27. DEVICE CONTROLLERS. For precise details about interrupt requests from device controllers, the specification for the controllers of interest should be consulted. In general, controllers generate interrupt requests when a previously initiated operation is completed.

2-28 MACHINE STACK OVERFLOW. The J500 processor architecture includes a 16 level pushdown stack. The stack is defined to be full when the bottom (16th) entry of the stack contains non-zero data. If a stack full condition occurs while interrupts are enabled, condition code 14 will be set true and an interrupt request will be made. The Branch-On-Condition instruction must be used to check the stack full condition code. Note that no stack full sensing occurs if interrupts are disabled.

SECTION 3

CENTRAL PROCESSOR

3-1. INTRODUCTION

The J500 contains a microprogrammed processor which not only executes MACROprogram instructions but also controls all of the peripherals. Because most of the control logic for peripherals is implemented by microcode instead of hardware logic, the J500 is smaller than other computers with equivalent capabilities.

The function of the microprogram is to control the flow of data between the major hardware elements of the J500. The most important elements are listed below:

- o Microprogram Memory - 4096 words by 56-bits.
- o Microprogram Control Logic.
- o Main Memory - 65536 words by 16-bits.
- o Register File - 256 words by 16-bits.
- o Arithmetic/Logic Unit (ALU) - 16-bits wide.
- o Input Multiplexor and Condition Bit.
- o Output Control Flip-Flops and Flag RAM.
- o Peripheral Input/Output Buffer Registers.
- o Two Programmable Sync/Async Communication Chips.
- o CRT Character Generator Memory.

A brief description of these hardware elements is given below.

3-2. MICROPROGRAM MEMORY

The microprogram memory consists of 28 PROMS that contain 4096 words of 56-bits. Each word in a microinstruction contains various fields that control the flow of data between different parts of the computer. The instructions in the microprogram are organized into groups of four, since it takes four to do a memory cycle.

There are really two programs in the microcode running in an interleaved fashion. A switch between the two programs occurs at the end of each memory cycle. One of the programs executes MACROprogram instructions and services microinterrupts from the peripherals. The other program fetches characters from the main memory for the CRT display and generates horizontal and vertical sync signals for the CRT.

3-3. MICROPROGRAM CONTROL LOGIC

The microprogram control logic handles the sequencing of instructions in the microprogram and implements the flow of data between the various hardware elements of the computer as dictated by the microprogram. Micro instructions are executed at the rate of one every 165 nanoseconds.

There is a separate microprogram step counter for each of the two programs. A four-word stack associated with each program step counter is used for saving the program counter on subroutine calls and interrupts.

Conditional branching of the microprogram is controlled by the condition-bit. There is also indirect jump, vectored jump, and subroutine jump and return capabilities. A special microinstruction decodes a MACROprogram instruction and does a vectored jump to the appropriate MACROinstruction emulation routine. Only one jump per memory cycle is permitted.

Special hardware that monitors status lines from the peripherals generates microinterrupts that cause a vectored jump to different peripheral service routines.

Other parts of the control logic generate signals that affect the main memory, ALU, register file, input multiplexor, condition-bit, output flip-flops, and peripheral buffer registers.

3-4. MAIN MEMORY

Main memory is the storage element for the operating system, application programs, data, and the CRT display. It consists of 64 dynamic MOS RAM chips that contain 65536 words by 16-bits. Each chip is organized as 16384 words by 1 bit. A memory address register (MAR) contains the address of the word being accessed. Dynamic RAM memories have the characteristic that they must be periodically "refreshed" to retain data. This is handled automatically by the CRT microprogram. Since a memory access requires four microinstruction cycles, the effective memory cycle time is 660 nanoseconds.

3-5. REGISTER FILE

The register file serves as a scratch pad memory for the microprogram. It consists of four fast bipolar static RAMS that contain 256 words by 16-bits. The register file is fast enough so that the processor can read a word, pass it through the ALU, and store the result back in the register file in one microinstruction cycle. The register file contains all of the MACROprogram registers as well as temporary storage used by the controllers for the various peripherals.

3-6. ARITHMETIC/LOGIC UNIT (ALU)

The Arithmetic/Logic Unit consists of four fast bipolar 74S281 ALU chips plus associated logic. Each chip is a 4-bit slice of the total 16-bit wide ALU. These chips are capable of performing 16 arithmetic and logic functions on an external input and an internal shift register. The external input can be either a register file word, data from one of the peripheral buffer registers, or data from main memory. The result of the ALU function can be routed to main memory, a register file location, or latched into the internal shift register shifted right or left one bit position. The condition-bit can be a right or left input to the shifter.

3-7. INPUT MULTIPLEXOR AND CONDITION-BIT

The input multiplexor selects one of 64 inputs or one of the 256 Flag RAM bits to be gated to the condition bit. Many of these input lines are status signals from various peripherals. The condition-bit is used to control branching of the microprogram.

3-8. OUTPUT FLIP-FLOPS AND FLAG RAM

The condition-bit can be routed to any of the 112 output flip-flops and 256 Flag RAM locations. The output flip-flops are generally control signals to various peripherals. The first 112 Flag RAM locations record the state of the corresponding output flip-flop; the rest are used as temporary storage by the microprogram.

3-9. PERIPHERAL BUFFER REGISTERS

The peripheral buffer registers are used to latch data and control signals going to peripherals, and capture data and status signals from peripherals while the microprogram is busy on other things. In the case of the disks, special hardware converts from 16-bit parallel to serial data on output, and serial to parallel on input. The CRT has a shift register that outputs the serial dots for a character.

3-10. PROGRAMMABLE SYNC/ASYNC COMMUNICATION CHIPS

Each communication line has a programmable communication chip (Signetics 2651) that serially outputs the bits of transmitted characters, and assembles the serial bits of received characters. The chip permits programming sync/async mode, baud rate, parity, character size, and number of stop bits.

3-11. CRT CHARACTER GENERATOR MEMORY

The dot patterns for the first 128 characters are stored in 3072 words by 8-bits of PROM memory. The dot patterns for the remaining 128 characters are stored in 3072 words by 8-bits of static MOS RAM memory. This latter memory can be loaded by the MACROprogram allowing the patterns for these characters to be programmed. Each character uses 20 words of memory corresponding to the 20 scan lines of a character. On a black/white CRT which has only 10 scan lines per character, half of the character generator memory is not used.

3-12. J500 MICROPROGRAM DESCRIPTION

There are actually two independent microprograms running in an interleaved fashion. A switch between the two programs occurs every four microinstructions (a memory cycle).

The CRT microprogram typically fetches a word from main memory and outputs it to the CRT character generator. It also generates horizontal and vertical sync signals required to produce the raster scan on the CRT. A portion of each horizontal scan line is blanked corresponding to the time when the beam is moving back to the left margin. During this time, the CRT microprogram does main memory "refresh" cycles and runs various timers. If a cursor is to be displayed on the next scan line, the microprogram substitutes a special character in the CRT refresh memory at the place where the cursor is to occur. Since a MACROprogram might be confused by this, the MACROprogram is turned off for the time it takes to do the scan line (approximately 62.5 microseconds). After the scan line is completed, the original character is restored in the refresh memory and the MACROprogram is resumed.

The other microprogram emulates MACROprogram instructions and handles microinterrupts. Microinterrupts are triggered by signals from peripherals and are serviced on a priority basis where the priority scale from highest to lowest is: cartridge disks, floppy disks, communication lines, keyboard, printer, and MACROprogram interrupt. When no microinterrupts are pending, the microprogram executes MACROinstructions. A microinterrupt causes a vectored jump to the section of microcode for handling the interrupting device. With two exceptions, microinterrupts can only occur at the end of processing a MACROinstruction or another microinterrupt. The exceptions are that a cartridge disk interrupt can occur any time, even during another microinterrupt, and microinterrupts can occur during shift/rotate and some I/O instructions. However, MACROinterrupts can only occur at the end of a MACROinstruction. The MACROprogram determines the device causing a MACROinterrupt by executing an INTA instruction which returns the device address of the highest priority device requesting an interrupt.

At the start of executing a MACROinstruction, the instruction word is stored in the Current Instruction Register (CIR), the MACROprogram counter (PC) has been advanced to the next instruction, and that word has been fetched from memory and stored in the Next Instruction Register (NIR). Because of this, all displacement fields in instructions are relative to PC+1. Somewhere during the execution of the instruction, the NIR is copied to the CIR, the PC is advanced, and the memory word addressed by the PC is fetched and stored in the NIR. Because instructions are fetched two words ahead, a word stored into PC+1 or PC+2 will not be executed as expected. Instead, the previous contents of the word will be used. On a JMP or JSR MACROinstruction the contents of the NIR are discarded, the PC is set to the location being jumped to, and that word is fetched and stored in the NIR. Then to complete the jump, the NIR is copied to the CIR, the PC is advanced, and that memory word is fetched and stored in the NIR. The microcode for each MACROinstruction ends by doing a special microinstruction that decodes the MACROinstruction in the CIR and does a vectored jump to the section of microcode which executes that MACROinstruction, or does a vectored jump to a microinterrupt service routine. MACROinterrupts are the lowest priority microinterrupt.

4-1. KEYBOARD.

The J500 International Keyboard is a data entry and control device that is available in other languages in addition to the American version shown in Figure 4-1. When a displayable character key is depressed, the character displayed is controlled by the SHIFT keys as shown in the illustration. Alphabetic characters are displayed in both upper and lower case. An "N" Key roll-over feature prevents conflicting inputs due to simultaneous depression of keys.

All keys on the keyboard are fully ASCII coded except for the REPEAT key. Depressing the REPEAT key and any alphanumeric or cursor function key will cause automatic repetition of the corresponding character or function at a rate of approximately 20 times per second until the key is released. This feature is a function of the Keyboard Controller.

4-2. CRT DISPLAY

The CRT display is arranged in 24 lines of 80 characters each. The character generator is capable of producing 256 individual characters of which 128 character codes are completely user programmable. The non-programmable are the standard upper and lower case ASCII characters (see Table 4-1). The characters are displayed in an 8 x 20 dot matrix in interleave mode or an 8 x 10 matrix in non-interleave mode. Each of the programmable characters is directly adjacent to neighboring characters allowing bar graphs and continuous curves to be displayed. The 1920 characters for the CRT display are stored in main memory starting at address Hex FC00 and are accessed the same as all other memory locations.

A blinking cursor can be moved to any character position on the screen or it can be moved completely off the screen. Special display attributes available to the user are double intensity, inverted video, underline, blanking and blinking.

4-3. AUDIO VOLUME, BRIGHTNESS AND CONTRAST CONTROLS

Two audio volume controls plus the CRT contrast and brightness controls are located on the front of the CRT. One audio control is for an audible "click" that is produced each time a key on the keyboard is depressed. The second audio control is for a 1/4 second audible alarm that can be sounded by the macro program by executing a DOB instruction. The volume of the "click", the volume of the alarm, the CRT brightness and the CRT contrast can be adjusted by the controls on the front on the CRT as shown in Figure 1-1.

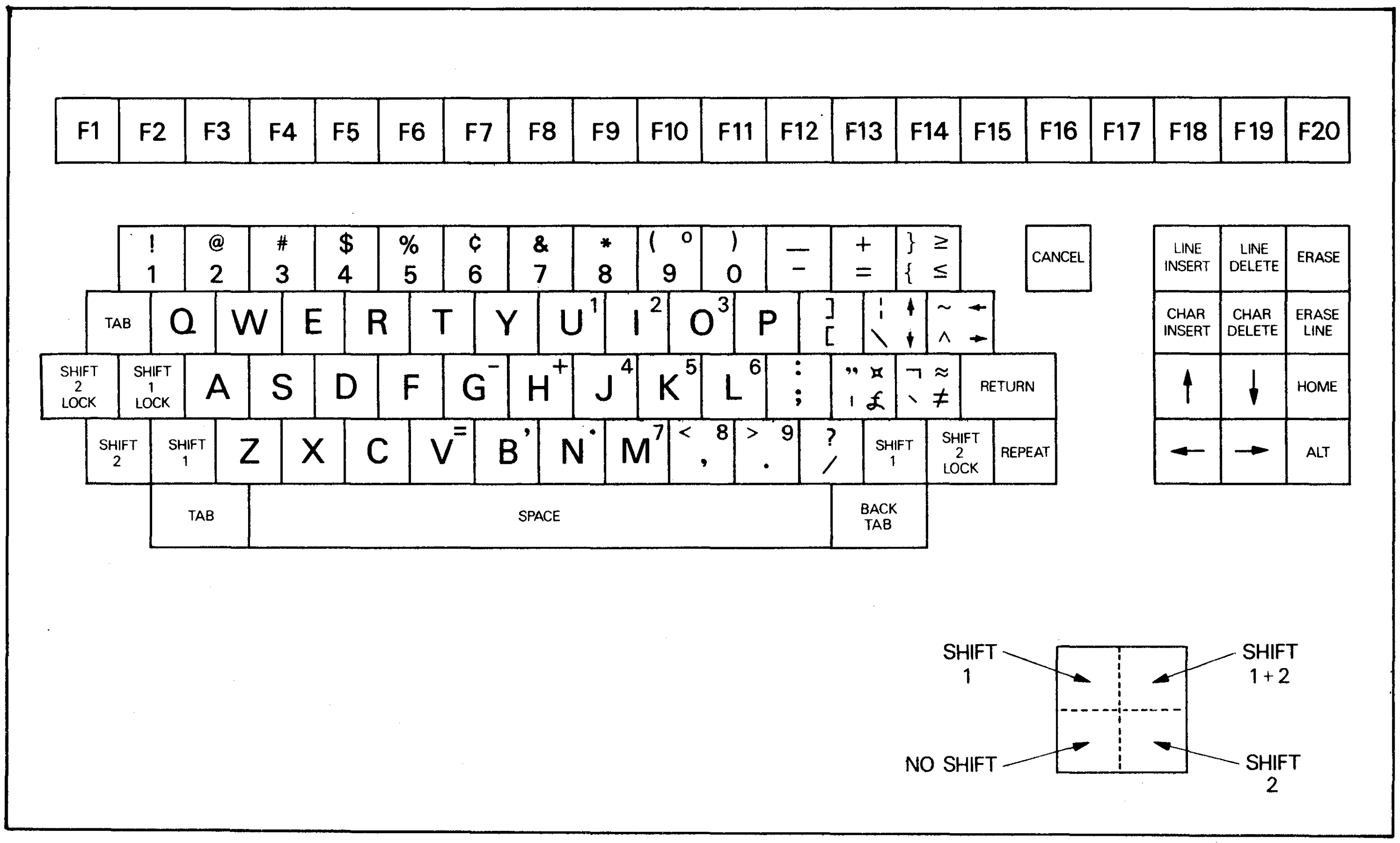


Figure 4-1. International Keyboard (American)

Table 4-1. Eight Bit ASCII Code

b8 →					0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
b7 →					0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
b6 →					0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
b5 →					0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
b4 ↓	b3 ↓	b2 ↓	b1 ↓	COLUMN → ROW ↓	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	0	0	0	↓	FIO	SP	o	@	P	‘	p	≤							
0	0	0	1	1	F9		!	1	A	Q	a	q	≥							F11
0	0	1	0	2	↑		"	2	B	R	b	r	≠							F12
0	0	1	1	3	ERAS		#	3	C	S	c	s	¢						≈	F13
0	1	0	0	4	ELIN		$	4	D	T	d	t							¤	F14
0	1	0	1	5	F7	LDEL	%	5	E	U	e	u							↑	F15
0	1	1	0	6	F8	LINS	&	6	F	V	f	v							→	F16
0	1	1	1	7	HOM	CDEL	’	7	G	W	g	w							↓	F17
1	0	0	0	8	CAN	CINS	(	8	H	X	h	x							←	F18
1	0	0	1	9	TAB	←	)	9	I	Y	i	y								F19
1	0	1	0	10	RET	→	*	:	J	Z	j	z								F20
1	0	1	1	11	F5	F6	+	;	K	[	k	{								
1	1	0	0	12	BACK TAB	F1	,	<	L	\	l	:								
1	1	0	1	13		F2	–	=	M	]	m	}								
1	1	1	0	14	TAF	F3	·	>	N	∧	n	~								
1	1	1	1	15	ALT	F4	/	?	O	–	o	¬								

Note: Those characters for which Bit 8 = 1 will not be displayed until the Programmable Character Generator has been loaded. See Paragraph 4-11. Software delivered with the J500 includes a program and data file to establish these characters.

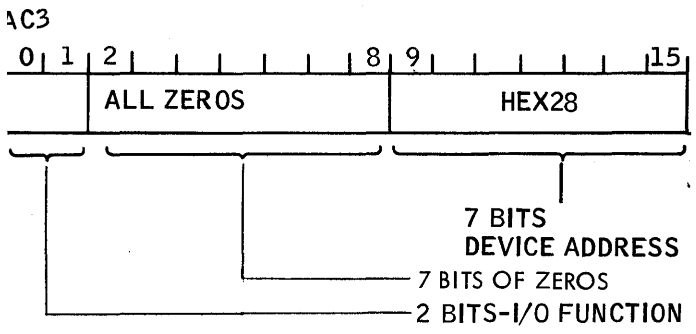
4-4. SPECIAL DISPLAY ATTRIBUTES

The display attribute for the first character on the CRT display is set by a DOC instruction as described in this section. The display attributes may be changed within the text by entering the appropriate character codes into the text. The attribute in effect at the end of one line is carried over to the beginning of the next line. Attribute codes that may be embedded in the text are listed below:

ATTRIBUTE	START HEX CODE	STOP HEX CODE
Half Intensity	11 or 1C	12 or 1D
Inverted Video	15	16
Underline	10 or 17	19 or 18
Blanking	1A	1B
Blinking	13	14

4-5. I/O INSTRUCTIONS

All I/O instructions require a 7-bit Device Code and a 2-bit Function Code to be located in Accumulator 3 (AC3). The format for the Device Code and Function Code is shown below:



I/O FUNCTION CODES (BITS 0 AND 1)

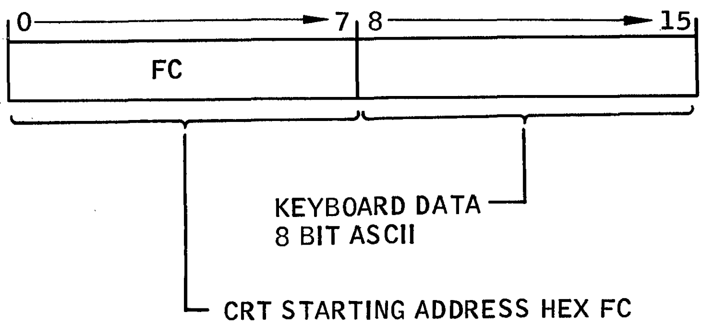
00 - NONE
01 - START - SETS BUSY, CLEARS DONE FLAG
10 - CLEAR - CLEARS BUSY AND DONE FLAGS
11 - NONE

NOTE

The normal Device Code for the CRT/Keyboard is Hex 28 but the controller will also respond to Hex 02.

Depressing a key on the keyboard will clear the CRT/Keyboard BUSY flag, set the DONE flag and unless this device is 'Masked Out' will cause a Macro Interrupt (see MSKO Instruction).

4-6. DATA IN A (DIA) INSTRUCTION. The DIA instruction performs the function Start, Clear or None as specified by AC3 and reads the keyboard Data Register and CRT Starting Address into AC0. The CRT Starting Address will always be Hex FC00. The word format is shown below:



4-7. DATA IN B (DIB) INSTRUCTION. The DIB instruction performs the function Start, Clear or None specified by AC3 and reads the value Hex 0001 into AC0.

4-8. DATA IN C (DIC) INSTRUCTION. The DIC instruction only performs the function Start, Clear or None as specified by AC3.

4-9. DATA OUT A (DOA) INSTRUCTION. The DOA instruction only performs the function Start, Clear or None as specified by AC3.

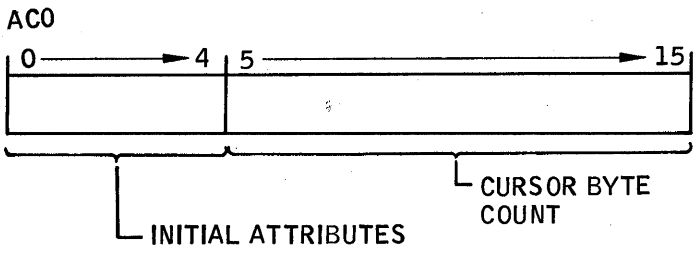
4-10. DATA OUT B (DOB) INSTRUCTION - WITHOUT START FUNCTION. The DOB instruction without Start function performs the Clear or None function as specified by AC3 and can sound a 1/4 second audible alarm. If AC0 bit 14 is a "1" an internal flag is set and if bit 14 is a "0" the flag is cleared. When this flag changes state from a "0" to a "1" the 1/4 second alarm is sounded.

4-11. DATA OUT B (DOB) INSTRUCTION - WITH START FUNCTION. The DOB with Start function performs the function Start as specified by AC3 and loads the CRT Programmable Character Generator. AC0 contains the first address of a 21 word block of memory. This block of memory contains the data used for loading the programmable Character Generator. The first word has the 8 bit character code located in the most significant 8 bits of the word. The most significant bit of this code must be a "1" to write into the Programmable Character Generator. The next 20 words contain the 8 bit patterns for the 20 horizontal lines that make up the character. These 8 bit patterns are located in the least significant 8 bits of the word. A "1" in the code means light and a "0" means dark.

NOTE

This instruction will not be completed until all operations to the communication lines, floppy drives and cartridge drives are completed. Micro and macro interrupts are serviced while waiting for these I/O devices to be completed.

4-12. DATA OUT C (DOC) INSTRUCTION. The DOC instruction performs the function Start, Clear or None as specified by AC3 and is used for positioning the cursor and specifying the initial attributes to be in effect for the first character of the display. The cursor will be displayed if the cursor byte count is in the range 000_{16} - $77F_{16}$ and will not be displayed if it is in the range 780_{16} - $7FF_{16}$. The word format and initial attributes are shown below:



INITIAL ATTRIBUTES

Bit 0 - Double Intensity

Bit 1 - Inverted Video

Bit 2 - Underline

Bit 3 - Blanking

Bit 4 - Blinking

4-13. NO I/O (NIO) INSTRUCTION. The NIO instruction performs the function Start, Clear or None as specified by AC3.

4-14. READ DONE (RDDNE) INSTRUCTION. The RDDNE instruction stores the state of the CRT/Keyboard Done flag into the CPU Done flag.

4-15. READ BUSY (RDBSY) INSTRUCTION. The RDBSY instruction stores the state of the CRT/Keyboard Busy flag into the CPU Busy flag.

4-16. MASK OUT (MSKO) INSTRUCTION. The MSKO instruction can be used to enable or disable macro interrupts for the CRT/Keyboard. Macro interrupts will be enabled if AC0 Bit 14 = 0 and disabled if AC0 Bit 14 = 1. AC3 is not used for this instruction.

SECTION 5. UNIVERSAL DISK CONTROLLER

5-1. INTRODUCTION.

The J500 Universal Disk Controller is a device which controls both the floppy disk and the cartridge disk. This allows for much similarity in the command protocol.

The J500 has the capability of controlling one floppy daisy chain with a Shugart compatible interface and one cartridge daisy chain with a Pertec compatible interface. Up to four devices can be connected on each daisy chain. All types of floppies can be mixed within their daisy chain (single-sided single-density, single-sided double-density, and double-sided double-density) but all cartridges must have the same data rate within their daisy chain.

5-2. I/O INSTRUCTIONS.

The Universal Disk Controller supports only three I/O nstructions. These instructions are DOA, DIA and DIA with Clear. All other instructions (DOB, DOC, NIO, DIB, DIC, RDBSY and RDDNE) are ignored and no interrupt is generated.

Each device has its own device code which must be in the right byte of AC3 for I/O operations. The device code is returned in the right byte of AC0 for Interrupt Acknowledge nstructions. The device codes in Hex are as follows:

DEVICE CODES

FLOPPY DRIVES	14 = FLOPPY DRIVE 0
	15 = FLOPPY DRIVE 1
	16 = FLOPPY DRIVE 2
	17 = FLOPPY DRIVE 3
CARTRIDGE DRIVES	18 = CARTRIDGE DRIVE 0
	19 = CARTRIDGE DRIVE 1
	1A = CARTRIDGE DRIVE 2
	1B = CARTRIDGE DRIVE 3

Although each floppy and each cartridge has a separate levice code, no overlap operations within a daisy chain are llowed. Only one operation in a daisy chain can be active t one time. If a second operation is attempted, no interrupt or this operation is generated and the program error status it is turned on in the status returned for the first operation.

-3. DATA OUT A (DOA) INSTRUCTION. The DOA instruction initiates a disk operation. The Device Code is placed in C3 and the address of a Command Control Block of words s in AC0. A Device Start is implied by this operation and n interrupt is generated (if interrupts are enabled) at the ompletion of each command. The Command Control Block ontains the following information:

ORD	DESCRIPTION
0	I/O COMMAND
1	OPTIONS
2	CYLINDER NUMBER
3	HEAD/PLATTER NUMBER
4	SECTOR NUMBER
5	SECTOR SIZE IN WORDS
6	NUMBER OF SECTORS TO PROCESS
7	NOT USED CURRENTLY
8	ADDRESS OF DMA BUFFER IN MEMORY

I/O COMMAND WORD (WORD 0). The I/O Command Word specifies the type of operation to be performed. Each bit of the word specifies a different command. Only one bit should be on at any given time with certain exceptions as noted.

HEX CODE	COMMAND	DESCRIPTION
0000	SELECT	No operation other than selecting a drive. It is necessary to wait for an interrupt after this command.
0001	SEEK	Move heads to specified cylinder. Will take into account any Track Offset specified in Word 1. This command can be used with a data transfer command (Read, Write, etc.). Will cause a 10 msec delay at completion of Seek to allow any offset to settle.
0002	RESTORE	Restore heads to cylinder 0, then seek to track specified in Word 2 if non-zero. Will take into account any Track Offset specified in Word 1. This command can be used with a data transfer command (Read, Write, etc.).
0004	READ	Read data starting at specified Cylinder, Head and Sector into memory at specified DMA address. Reads specified number of sectors and performs auto seek if required.
0008	Not Used	
0010	WRITE	Write data from specified memory DMA address to disk starting at specified Cylinder, Head and Sector. The prior sector header is read on the cartridge to verify positioning before writing. Writes specified number of sectors and performs auto seek if required.
0020	WRITE DELETED DATA MARK	This command is only used for floppies. Same as write except that a special Deleted Data mark is written in the floppy disk sector. Necessary for IBM compatible floppy disks. Performs auto seek if required.
0040	HEADER CHECK INHIBIT WRITE	Normal write for floppies. Same as write for cartridges except that the prior sector header is not verified before writing. Performs auto seek if required.
0080	INITIALIZE	This command used only for floppies. Initialize specified track (cylinder, head) by writing sector headers, gaps and data fields for entire track. Performs auto seek if required. Refer to floppy format requirements.

HEX CODE	COMMAND	DESCRIPTION
0100	Not Used	
0200		
0400		
0800		
1000		
2000		
4000		
8000		

OPTION WORD (WORD 1). The Option Word is used during Seek and Restore operations for the cartridge and data operations for the floppy. The specific bits are defined below. The most significant three bits are reserved for use by the macro program and are not sampled by the controller.

HEX CODE	COMMAND	DESCRIPTION
0001	TRACK OFFSET PLUS	This bit is used for cartridge Seek and Restore operations only. Moves the head off track toward the spindle.
0002	TRACK OFFSET MINUS	This bit is used for cartridge Seek and Restore operations only. Moves the head off track away from the spindle.
0004	DOUBLE DENSITY	This bit is used only for floppy data operations. Will force the floppy to read and write double density.

All other bits of this word are not used.

NOTE

If both Track Offset bits are on, no head offset will take place but the gain of the read amplifier is lowered. No offsets should be on when writing as the data will be written offset.

The two offset bits are sampled only when a Seek or Restore command is issued or if an auto seek to a different cylinder is required before a data operation. Whenever the offset bits are changed, the Seek bit should be turned on in the command word to allow an extra 10 msec of head settling time. Failure to turn on the seek bit might result in a read or write operation taking place while the head is still moving.

CYLINDER NUMBER (WORD 2). This word contains the number of the cylinder where the read/write head is to be placed. The following is a decimal list of legal cylinder numbers for the different disk types.

DATA BITS	CYLINDERS	DRIVE TYPE
9-15	0-76	All Floppies
8-15	0-202	Cartridge-6 Megabyte
7-15	0-405	Cartridge-12 Megabyte

HEAD/PLATTER NUMBER (WORD 3). This word specifies which platter and head on which a data operation is to take place. This word is sampled only on data operations.

HEX CODE	FUNCTION	DESCRIPTION
0001	HEAD BIT	This bit when true will select the upper head. This bit when false will select the correct side of a single-sided floppy.
0002	PLATTER BIT	This bit when true will select the upper platter. Only used for cartridge drives.
0004	EXTENSION PLATTER PAIR SELECT BIT	This bit when true will select the lower pair of platters on 24 megabyte cartridge drives.

All other bits of this word are not used.

SECTOR NUMBER (WORD 4). This word contains the number of the first sector to be processed. The following is a decimal list of legal sector numbers as currently used by System II.

DATA BITS	SECTORS	DRIVE TYPE
11-15	1-26	Floppy-Single Density
12-15	1-15	Floppy-Double Density
12-15	0-11	All Cartridges

SECTOR SIZE IN WORDS (WORD 5). This word specifies the number of words to be written or read from one sector during a data operation. The following is a decimal list of legal sector sizes for System II formats.

DATA BITS	WORDS	DRIVE TYPE
9-15	64	Floppy-Single Density
7-15	256	Floppy-Double Density
7-15	256	All Cartridges

NUMBER OF SECTORS TO PROCESS (WORD 6). This word specifies the number of sectors to be read or written during a data operation. This number must be greater than zero but not greater than the maximum number of sectors per track. When doing multi-sector operations, the last sector number processed must be greater than the first sector processed. For example, starting Sector is 1 and Number of Sectors to Process is 26 is correct for single-density floppies, but if Starting Sector is 2 and Number of Sectors to Process is 26 is incorrect.

WORD 7. Reserved for future use.

ADDRESS OF DMA BUFFER (WORD 8). This word contains the starting address of the block of memory to be read from or written into. The size of this block of memory is determined by the number of words per sector times the number of sectors to process.

5-4. DATA IN A (DIA) INSTRUCTION - WITHOUT CLEAR. The DIA instruction is used to request status from the device specified by the Device Code in AC3. If no device on the daisy chain is busy, the device for which the status is being requested is selected and current status for that drive is deposited in AC0. If the drive for which the status is being requested is busy, current status is given. However, if another drive on the daisy chain is busy, the status that is given is not "current" status, but instead is the status for that drive the last time it was updated by the microprogram. The status is updated at various points during an operation and at its completion, but will not be updated again until a new command is given.

The status from the disk is put into CPU register AC0. The bits in the status word have the following meaning. Any combination of bits may occur.

BIT	MEANING
0001	Daisy Chain Busy. This bit is true if any device on the selected daisy chain is busy.
0002	Not Used (always zero).
0004	Write Protected. Selected drive is write protected.
0008	CRC Error. CRC error was detected in Data Field.
*0010	Deleted Data Mark. This bit is true if the selected sector contained a deleted data mark.
0020	Drive Not Ready. This bit is true when the selected drive is not ready (door open, not up to speed).
*0040	Heads Unloaded. This bit is true when the selected drives head is not loaded.
0080	Program Error. This bit is true when a command is issued and a command is still in progress.
0100	Format Error. This bit is true if an illegal sector number is issued or when a sector overrun condition occurs.
*0200	Disk Change. This bit is true if the drive went not ready since the last command.
0400	Seek Error. Status from the drive or an illegal cylinder number was issued.
0800	Sector Not Found. This bit is true for an illegal sector number, header CRC error or sector length mismatch.
1000	DMA Overrun Error. Indicates a data channel overrun condition.
*2000	Header Sector Length Mismatch. Header sector length does not agree with command word sector length.
*4000	Double Sided Diskette. Indicates addressed floppy is double-sided drive.
8000	Not Used. Always zero.

*These bits are used only for floppy status and are always zero for cartridge.

5-5. DATA IN A (DIA) - WITH CLEAR. A Clear bit (most significant bit) in AC3 is only recognized with a DIA instruction. Normal status is placed in AC0, then any operation or pending interrupt for the device specified by AC3 will be canceled. If a device on the daisy chain other than the device specified by AC3 is busy, the Clear has no effect. If Clear is issued while an operation is in progress, the operation will be terminated immediately.

5-6. WRITING CONSIDERATIONS.

Care must be taken when using the Header Check Inhibit Write operation on cartridge drives and the Initialize operation on floppy drives. These operations do not verify head position before writing. If the head was positioned incorrectly, data could be destroyed. The cartridge can "get lost" by spinning the drive down and then up again while the floppy can get lost by having its door opened and then closed.

5-7. ADDITIONAL FLOPPY CONSIDERATIONS

The floppy controller will verify the complete header before any read or write operation. The side bit, track, sector size, density and CRC of the header must be correct before any read or write will take place. The controller will seek to the correct track and select the correct head if not positioned correctly before the data operation. The controller will check the header CRC as well as the data CRC. When writing a sector, the controller will compute the CRC for that sector and write it following the data field per IBM spec.

To decrease disk head and media wear, each floppy will have an individual head load timer. When any command is received, the selected drive will load its head and stay loaded for two seconds. Unselected units are not affected.

5-8. INITIALIZING FLOPPIES

The J500 Floppy Controller has the capability of initializing blank diskettes. Each track will be written from beginning to end. All data and clocks for the initialization process will be stored in memory and transferred to the disk on a DMA basis. The block of memory should be set up with alternating words of data and clocks. This memory block must include all headers, check characters, gap information, address marks and clock patterns for the specific track being initialized. The standard check character is represented by the polynomial:

$$G(X) = 1 + X^5 + X^{12} + X^{16}$$

Storing the initialization in memory allows the floppy to be initialized to any format. The only hardware restriction imposed upon the format is that in single density, all address, data, deleted data and index marks be preceded by at least 10 bytes of all 1's and exactly 6 bytes of 0's and that the clock patterns are correct.

Table 5-1 shows the recommended IBM 3740 format for single density floppies. This block of memory is to be loaded prior to giving the initialize command.

For double density floppies, the address marks (4 bytes per mark) must be preceded by at least 20 bytes of Hex 4E and exactly 12 bytes of 0's. These address mark conventions are to IBM spec.

Table 5-2 shows the recommended 15 sector format for double density floppies. IBM formats are available for different sector sizes. This block of memory is to be loaded prior to giving the format command.

Table 5-1. Diskette Initializing Format - Single Density

Word Offset (Decimal)	Contents (Hex)	Remarks
0, 2, . . ., 38	FFFF	20 WORDS
1, 3, . . ., 39	FFFF	
40, 42, 44	0000	3 WORDS
41, 43, 45	FFFF	
46	FCFF	FC = DATA INDEX MARK
47	D7FF	D7 = CLOCK INDEX MARK
48, 50, . . ., 70	FFFF	12 WORDS
49, 51, . . ., 71	FFFF	
72	FF00	
73	FFFF	
74, 76	0000	2 WORDS
75, 77	FFFF	
78	00FE	FE = DATA ID MARK
79	FFC7	C7 = CLOCK ID MARK
80	XX00	XX = CYLINDER NO., 00 = HEAD NO. (SINGLE SIDE)
81	FFFF	
82	XX00	XX = SECTOR NO. (01 = FIRST SECTOR)
83	FFFF	
84	XXXX	CRC FOR PRECEDING 5 DATA BYTES
85	FFFF	
86, 88, . . . 94	FFFF	5 WORDS
87, 89, . . . 95	FFFF	
96	FF00	
97	FFFF	
98, 100	0000	2 WORDS
99, 101	FFFF	
102	00FB	FB = DATA MARK FOR DATA
103	FFC7	C7 = CLOCK MARK FOR DATA
104, 106, . . ., 228	0000	63 WORDS
105, 107, . . ., 229	FFFF	
230	XXXX	SOFTWARE CHECKSUM
231	FFFF	
232	XXXX	CRC FOR PRECEDING 129 DATA BYTES
233	FFFF	
234, 236, . . ., 258	FFFF	13 WORDS
235, 237, . . ., 259	FFFF	
260	FF00	
261	FFFF	
262 - 449	REPEAT 74 - 261	SECTOR 2
450 - 637	REPEAT 74 - 261	SECTOR 3
638 - 825	REPEAT 74 - 261	SECTOR 4
826 - 1013	REPEAT 74 - 261	SECTOR 5
1014 - 1201	REPEAT 74 - 261	SECTOR 6
1202 - 1389	REPEAT 74 - 261	SECTOR 7
1390 - 1577	REPEAT 74 - 261	SECTOR 8
1578 - 1765	REPEAT 74 - 261	SECTOR 9
1766 - 1953	REPEAT 74 - 261	SECTOR 10
1954 - 2141	REPEAT 74 - 261	SECTOR 11
2142 - 2329	REPEAT 74 - 261	SECTOR 12
2330 - 2517	REPEAT 74 - 261	SECTOR 13
2518 - 2705	REPEAT 74 - 261	SECTOR 14
2706 - 2893	REPEAT 74 - 261	SECTOR 15
2894 - 3081	REPEAT 74 - 261	SECTOR 16
3082 - 3269	REPEAT 74 - 261	SECTOR 17
3270 - 3457	REPEAT 74 - 261	SECTOR 18
3458 - 3645	REPEAT 74 - 261	SECTOR 19
3646 - 3833	REPEAT 74 - 261	SECTOR 20
3834 - 4021	REPEAT 74 - 261	SECTOR 21
4022 - 4209	REPEAT 74 - 261	SECTOR 22
4210 - 4397	REPEAT 74 - 261	SECTOR 23
4398 - 4585	REPEAT 74 - 261	SECTOR 24
4586 - 4773	REPEAT 74 - 261	SECTOR 25
4774 - 4959	REPEAT 74 - 259	SECTOR 26
4960, 4962, . . ., 5306	FFFF	174 WORDS
4961, 4963, . . ., 5307	FFFF	GAP AT END OF TRACK

Table 5-2. Diskette Initializing Format - Double Density

Word Offset (Decimal)	Contents (Hex)	Remarks
0, 2, . . ., 78	4E4E	40 WORDS
1, 3, . . ., 79	FFFF	
80, 82, . . ., 90	0000	6 WORDS
81, 83, . . ., 91	FFFF	
92	C2C2	DATA INDEX MARK
93	F7F7	CLOCK INDEX MARK
94	C2FC	DATA INDEX MARK
95	F7FF	CLOCK INDEX MARK
96, 98, . . ., 144	4E4E	25 WORDS
97, 99, . . ., 145	FFFF	
146, 148, . . ., 156	0000	6 WORDS
147, 149, . . ., 157	FFFF	
158	A1A1	DATA ID MARK
159	FBFB	CLOCK ID MARK
160	A1FE	DATA ID MARK
161	FBFF	CLOCK ID MARK
162	XXYY	XX = CYLINDER NO., YY = HEAD NO.
163	FFFF	
164	XX00	XX = SECTOR NO. (01 = FIRST SECTOR)
165	FFFF	
166	XXXX	CRC FOR PRECEDING 8 DATA BYTES
167	FFFF	
168, 170, . . ., 188	4E4E	11 WORDS
169, 171, . . ., 189	FFFF	
190, 192, . . ., 200	0000	6 WORDS
191, 193, . . ., 201	FFFF	
202	A1A1	DATA MARK FOR DATA
203	FBFB	CLOCK MARK FOR DATA
204	A1FB	DATA MARK FOR DATA
205	FBFF	CLOCK MARK FOR DATA
206, 208, . . ., 714	0000	255 WORDS
207, 209, . . ., 715	FFFF	
716	XXXX	SOFTWARE CHECKSUM
717	FFFF	
718	XXXX	CRC FOR PRECEDING 516 DATA BYTES
719	FFFF	
720, 722, . . ., 802	4E4E	42 WORDS
721, 723, . . ., 803	FFFF	
804 - 1461	REPEAT 146 - 803	SECTOR 2
1462 - 2119	REPEAT 146 - 803	SECTOR 3
2120 - 2777	REPEAT 146 - 803	SECTOR 4
2778 - 3435	REPEAT 146 - 803	SECTOR 5
3436 - 4093	REPEAT 146 - 803	SECTOR 6
4094 - 4751	REPEAT 146 - 803	SECTOR 7
4752 - 5409	REPEAT 146 - 803	SECTOR 8
5410 - 6067	REPEAT 146 - 803	SECTOR 9
6068 - 6725	REPEAT 146 - 803	SECTOR 10
6726 - 7383	REPEAT 146 - 803	SECTOR 11
7384 - 8041	REPEAT 146 - 803	SECTOR 12
8042 - 8699	REPEAT 146 - 803	SECTOR 13
6700 - 9357	REPEAT 146 - 803	SECTOR 14
9358 - 10,015	REPEAT 146 - 803	SECTOR 15
10016, 10018, . . ., 11014	4E4E	500 WORDS
10017, 10019, . . ., 11015	FFFF	GAP AT END OF TRACK

6-1. INTRODUCTION.

The J500 has a microprogrammed communications controller that supports two independent communications lines with complete modem controls. There is a separate RS-232 connector on the rear of the J500 for each communication line. These connectors are designed to be connected with the standard cable to a modem. If it is desired to connect directly to a terminal (bypassing modems), a special adapter is required.

All of the options below are programmable for each of the lines.

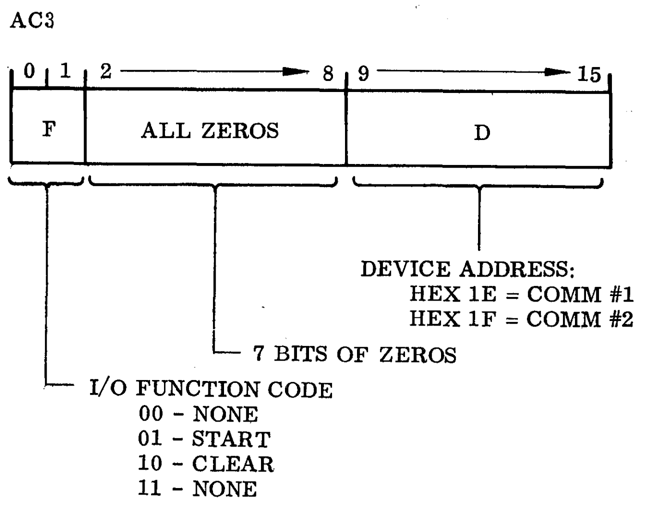
- Synchronous or Asynchronous Mode
- 16 Baud rates from 50 to 19200 Hz
- Internal or External Clocks set independently for the transmitter and receiver
- Character size of 5, 6, 7 or 8 bits
- Even or odd parity, enabled or disabled
- Stop bits of 1, 1.5 or 2 in Asynchronous Mode
- One or two sync characters and the DLE character in Synchronous Mode
- Transparent or non-transparent Binary Synchronous Mode
- Automatic echo in Asynchronous Mode
- Send a BREAK character in Asynchronous Mode
- Local or remote loop-back test mode
- Set modem signals: Data Terminal Ready, Request To Send and Line Busy. Test modem signals: Data Set Ready, Carrier Detect and Clear To Send.

Each of the communications lines consists of a Signetics 2651 Programmable Communications Interface (PCI) chip along with the associated RS-232 line drivers and receivers. The communications controller can be operated via macroprogram interrupts or by polling and reading the 2651 status register.

The controller can be operated in a single character at a time mode where an interrupt is triggered whenever a character is transmitted or received, or in a DMA mode where characters to be transmitted or received are stored in circular buffers in memory. In order to run the communications line at a high baud rate, it is necessary to use the DMA interface for buffering data on input and output.

6-2. I/O INSTRUCTIONS.

All I/O instructions require the simultaneous use of CPU accumulators AC0 and AC3. AC3 shown below, contains the I/O address and a function code.



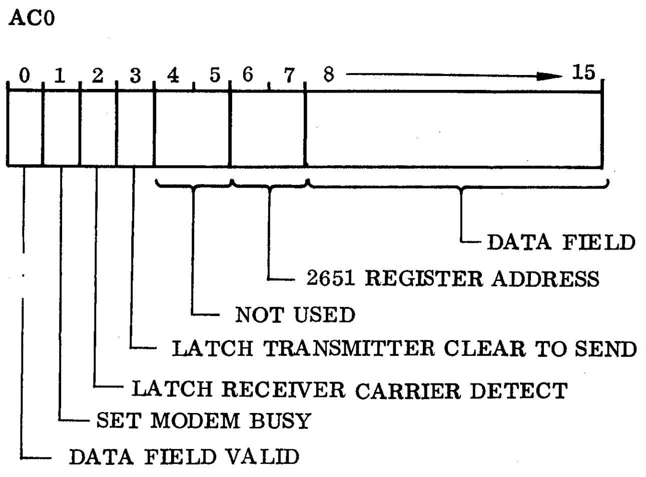
START - CLEAR CURRENT INTERRUPT AND ENABLE FURTHER INTERRUPTS.

CLEAR - CLEAR CURRENT INTERRUPT AND INHIBIT FURTHER INTERRUPTS

NOTE

If both bit 0 and 1 of AC3 are set, the result is the same as if only bit 0 was set.

6-3. DATA OUT A (DOA) INSTRUCTION. The DOA instruction is used to load the registers in the 2651 chip as described below:



Bit 0 If set, the data field is sent to the 2651 register specified by bits 6-7. Otherwise, the data field is ignored. In either case, the register address in bits 6-7 is saved for use with the next DIA instruction.

NOTE

Bits 1, 2, and 3 are processed only if Bit 0 = 1 and 2651 Reg 3 (Command Register) is referenced.

Bit 1 If set, the BUSY signal is turned on. On some modems, this causes anyone calling in to the computer to get a busy signal.

Bit 2 If set, the Carrier Detect signal to the 2651 chip is turned on when the Carrier Detect Signal (DCD) from the modem comes on, and it is kept on until a DOA instruction with Bit 2=0 is executed while the modem Carrier Detect signal is off. Otherwise, the Carrier Detect signal to the 2651 chip turns on and off when the modem Carrier Detect (DCD) turns on and off. The use of this feature is required when it is necessary to see a partial character received just before the carrier goes off since the Receiver Shift Register in the 2651 chip stops when the carrier goes off.

Bit 3 If set, the Clear to Send signal to the 2651 chip is turned on when the Clear to Send signal (CTS) from the modem comes on, and it is kept on until a DOA instruction with Bit 3=0 is executed while the modem Clear to Send signal is off. This feature is necessary in asynchronous mode in certain applications where the Clear to Send signal may turn off while data is being transmitted. If the Clear to Send input to the 2651 chip goes off while the chip is transmitting a stop bit, the chip will hang in the start bit state which causes a long space condition on the transmit data line. This will be treated as a BREAK character by the receiving device.

Bits 6,7 These bits indicate a register in the 2651 chip to be loaded from Bits 8-15.

Bit 6	Bit 7	Register
0	0	Transmitter Holding Register
0	1	SYN1/SYN2/DLE Registers
1	0	Mode Registers 1 and 2
1	1	Command Register

Bits 8-15 These bits are the data field for each of the 2651 registers described below.

Transmitter Holding Register

The Transmitter Holding Register in the 2651 chip contains the next character to be transmitted. This character is copied to the Transmitter Shift Register (and an interrupt is triggered) at the start of transmission of the last bit of the current character in the Shift Register. If the character is shorter than 8 bits, it must be right justified in the data field (bits 8-15) of AC0. If bit-13 of AC0 is set after reading the 2651 Status Register, the chip is ready for the Transmitter Holding Register to be loaded. In synchronous mode, the program must load a new character into the Transmitter Holding Register within n bit times (where n is the character size) after the Transmitter Holding Register empty interrupt or a fill character will be sent. In asynchronous mode, delays between characters are permissible. The Transmitter Holding Register should be loaded directly only if the transmit DMA feature is not being used.

SYN1/SYN2/DLE Registers

In synchronous mode the SYN1, SYN2, and DLE registers must be loaded prior to using the com line for communications to specify the characters to be used for synchronization and the characters to be used for fill when the transmitter is idle. The registers are loaded by three successive DOA instructions all referring to 2651 register-1, where the first loads SYN1, the second SYN2, and the third DLE. The 2651 chip contains an internal sequencer that addresses these three registers. The sequencer is reset to the SYN1 register by a NIO instruction which resets the 2651 chip, by a read or load command register operation, or after loading the DLE register. If the chip is being operated in single sync character mode, the same character should be loaded into SYN1 and SYN2. In single sync mode, SYN1 is used for synchronization, SYN1 is used for fill in non-transparent mode, and DLE-SYN1 is used for fill in transparent mode. In double sync mode, SYN1-SYN2 is used for synchronization, SYN1-SYN2 is used for fill in non-transparent mode, and DLE-SYN1 is used for fill in transparent mode. Both the transmitter and receiver should be disabled, and both the transmitter and receiver DMA control block addresses should be set to zero when these registers are loaded.

Mode Registers

The 2651 Mode registers must be loaded by two successive DOA instructions both referring to 2651 register-2, where the first loads Mode register-1, and the second loads mode register-2. The 2651 chip has an internal sequencer that addresses these two registers. The sequencer is reset to Mode register-1 by a NIO instruction which resets the 2651 chip, by a read or load command register operation, or after loading Mode register-2. Both the transmitter and receiver should be disabled, and both the transmitter and receiver DMA control block addresses should be set to zero when these registers are loaded.

Mode Register 1:

This register indicates synchronous/asynchronous mode, character length, and parity.

In asynchronous mode, it is possible to specify that the clock rate is divided by a factor of 16 or 64 to get the baud rate. However, this applies only if the external clock input is selected by Mode register-2.

If parity is enabled, a parity bit is added to the transmitted character and the receiver performs a parity check on incoming data. Odd or even parity can be selected.

The character size may be set to 5, 6, 7, or 8 bits. However, the character size does not include the parity bit, start bit, or stop bits in asynchronous mode.

In asynchronous mode, the number of stop bits may be 1, 1.5 or 2. If the internal transmitter clock is selected or the baud rate division factor is 1, 1.5 stop bits default to 2 stop bits on transmit.

In synchronous mode, one or two SYN characters can be programmed for synchronization and character fill when the transmitter is idle. The chip can also be programmed for transparent or non-transparent mode as in the IBM Binary Synchronous Communications protocol.

The meaning of the Mode register-1 bits as they appear in CPU register AC0 is as follows:

Bits	Meaning
8-9	In asynchronous mode, number of stop bits: 00 = invalid 01 = 1 stop bit 10 = 1-1/2 stop bits 11 = 2 stop bits In synchronous mode, number of sync chars: 00 = double sync, non-transparent mode 01 = double sync, transparent mode 10 = single sync, non-transparent mode 11 = single sync, transparent mode
10	Parity: 0 = odd, 1 = even
11	Parity control: 0 = disabled, 1 = enabled
12-13	Character length: 00 = 5 bits 01 = 6 bits 10 = 7 bits 11 = 8 bits
14-15	Synchronous/Asynchronous mode: 00 = synchronous, clk rate 01 = asynchronous, clk rate 10 = asynchronous, clk/16 rate (ext clock only) 11 = asynchronous, clk/64 rate (ext clock only)

Mode Register 2:

This parameter sets the baud rate. An external or internal clock may be selected independently for the transmitter and receiver. Any of 16 internal baud rates may be selected. If internal clocks are selected for both the transmitter and receiver, the same baud rate is used for both. If separate baud rates for the transmitter and receiver are required, external clocks must be selected and the proper clock signal must be supplied by the modem. Regardless of whether an external or internal clock is selected for the transmitter, the transmitter clock is output on pin-24 of the RS-232 connector.

When the 2651 chip is driven by a 5.0688 MHz clock input the internal baud rates have zero error except at 134.5, 2000, and 19200 baud, which have errors of +0.016%, +0.235%, and +3.125% respectively.

The meaning of the Mode register-2 bits as they appear in CPU Register AC0 is as follows:

Bits	Meaning
8-9	Not used (zero)
10	Transmitter clock: 0 = external, 1 = internal
11	Receiver clock: 0 = external, 1 = internal
12-15	Internal baud rate:

0000 = 50 baud	1000 = 1800 baud
0001 = 75	1001 = 2000
0010 = 110	1010 = 2400
0011 = 134.5	1011 = 3600
0100 = 150	1100 = 4800
0101 = 300	1101 = 7200
0110 = 600	1110 = 9600
0111 = 1200	1111 = 19200

Command Register

The 2651 Command Register controls the transmitter enable and receiver enable in the 2651 chip, as well as the Data Terminal Ready (DTR) and Request To Send (RTS) signals to the modem.

Disabling the receiver causes a partial character in the receiver shift register (if any) to be lost.

Disabling the transmitter will cause a partial character in the transmitter shift register (if any) to be transmitted prior to turning off the transmitter.

The 2651 can operate in one of four sub-modes within each major mode (synchronous or asynchronous) depending on the options selected in the Mode and Command registers. The submodes are: normal, auto echo (asynchronous) or SYN/DLE stripping (synchronous), local loop back, and remote loop back.

If asynchronous auto echo mode is selected, received data is automatically copied to the transmitter holding register and then retransmitted. The receiver must be enabled, but the transmitter need not be enabled. The normal CPU to transmitter link is disabled in this mode, but the receiver operates normally. In this mode, the receiver clock is also used as the transmitter clock. The 2651 must be taken out of auto echo mode in order to transmit data from the CPU.

In synchronous mode with automatic SYN/DLE stripping, the exact action taken depends on whether the 2651 is in transparent mode. In non-transparent single SYN mode, characters in the data stream matching SYN1 are discarded. In non-transparent double SYN mode characters in the data stream matching SYN1, or SYN2 if immediately preceded by SYN1 are discarded; however, only the first SYN1 or a SYN1-SYN1 pair is discarded. In transparent mode, characters in the data stream matching DLE, or SYN1 if immediately preceded by a DLE are discarded; however, only the first DLE or a DLE-DLE pair is discarded.

In local Loop-Back Mode, the transmitter output is connected to the receiver input. Data Terminal Ready (DTR) is connected to Data Carrier Detect (DCD), and Request to Send (RTS) is connected to Clear to Send (CTS). The receiver clock is the same as the transmitter clock. Any data transmitted comes back in through the receiver. The Receiver Enable bit in the command register is ignored, but the Request to Send and Transmitter Enable bits in the command register must be on for any data to go through. This mode is equivalent to putting a special loop-back plug on the RS-232 connector on the back of the computer.

In Remote Loop-Back Mode, the receiver input is connected to the transmitter output so that any data received is automatically retransmitted. The transmitter clock is the same as the receiver clock. The Transmitter Enable bit in the command register is ignored, but the Receiver Enable bit must be on in order for any data to be retransmitted. The following error conditions will always be set in this mode: parity error, overrun error, framing error. This mode is equivalent to putting a loop-back plug on the RS-232 connector on the modem.

In synchronous mode, turning on the receiver causes the chip to go into a "hunt" mode looking for a synchronization character sequence. In this mode, as data is shifted into the Receiver Shift Register a bit at a time, the contents of the register are compared to the SYN1 register. When the two registers match, the hunt mode is terminated and character assembly mode begins. If double sync mode is programmed, the first character after SYN1 must be SYN2 or the chip returns to hunt mode. Note that the sequence SYN1-SYN1-SYN2 does not achieve synchronization. After synchronization is achieved the chip continues to assemble characters and transfer them to the Receiver Holding Register. When it becomes necessary to re-synchronize, the receiver must be disabled and then re-enabled.

It is permissible to enable the receiver before the Carrier Detect from the modem is obtained since the chip will not assemble data until Carrier Detect is present. Similarly, it is permissible to enable the transmitter before Clear to Send from the modem is on since the chip will not transmit data until Clear to Send is present.

The meaning of the Command register bits as they appear in CPU register AC0 is as follows:

Bits	Meaning
8-9	Operating mode: 00 = Normal operation 01 = In asynchronous mode, auto echo mode; SYN/DLE stripping in synchronous mode 10 = Local loop back 11 = Remote loop back
10	Request to send: 0=off, 1=on
11	Reset error: 0=ignore, 1=reset the following error flags in the 2651 status register: framing error/sync character detected, receiver overrun, parity error/DLE character detected
12	In asynchronous mode, send a BREAK character until next command. In synchronous mode, send a DLE character before the character in the transmitter holding register.
13	Receiver enable: 0=off, 1=on.
14	Data terminal ready: 0=off, 1=on.
15	Transmitter enable: 0=off, 1=on.

6-4. DATA OUT B (DOB) INSTRUCTION. The DOB instruction is used to specify the location of the transmitter DMA control block in memory.

```
DOB    where AC3:  bits 9-15 = device code
             AC0:  address of transmit control block
```

An address of zero implies no control block and thus no data to be transmitted. This should be done when the device is closed.

This instruction must be executed to initialize DMA mode and when the entries in the DMA control block are changed by the CPU after the buffer has gone empty. As long as the buffer contains data, changes in the control block will be recognized by the controller.

The DMA control block describes a circular buffer which contains the data to be transmitted. The buffer is defined by an initial memory address (TBA), a length (TBL), an input pointer (TIP), and an output pointer (TOP). The TIP and TOP pointers are both relative to the origin of the buffer. The TIP pointer defines where the next data entry is to be put into the buffer by the CPU, and the TOP pointer defines where the next data entry is to be taken out by the controller. The buffer is called circular because when either TIP or TOP gets equal to TBL it is reset to zero. Thus, the buffer contains data from TOP up to TIP if TIP > TOP, or from TOP up to TBL and from the beginning up to TIP if TIP < TOP.

When DMA output is active, the CPU is putting data into the buffer while advancing the TIP pointer, and the controller is chasing along behind taking data out of the buffer while advancing the TOP pointer. The buffer is empty when TOP catches TIP. The buffer is full when TIP catches TOP-1. The buffer can be treated as a normal linear non-wraparound type buffer if the TOP pointer never wraps around.

A data entry in the transmitter DMA buffer consists of a character to be transmitted which must be stored right justified in bits 8-15 of the word.

No data will be transmitted until the transmitter is enabled by a command to the 2651. As long as there is data in the circular buffer, the controller will continue to feed characters to the 2651 Transmitter Holding Register whenever it is empty. After the buffer goes empty and the last character has been output from the Transmit Shift Register, the Transmitter Empty status bit will come on, an interrupt will be triggered, and the Transmit Data Line will go into a MARK state in asynchronous mode or begin transmitting SYN1 characters in synchronous mode (DLE SYN1 characters in transparent mode). If more data is then put into the buffer, and a DOB instruction is issued, the controller will resume transmission.

There is an optional command which can be sent to the 2651 chip when the buffer goes empty. This can be used, for example, to turn off the transmitter enable.

The transmit control block contains the following parameters.

Word	Name	Meaning
0	TBA	Address in memory of transmitter circular DMA buffer.
1	TBL	Length in words of transmit buffer.
2	TIP	Offset relative to the beginning of buffer where the CPU is to put the next data entry. This parameter must be updated by the CPU whenever data is put into the buffer.
3	TOP	Offset relative to beginning of buffer where controller is to pick up next data entry to be transmitted. This parameter is updated by the controller each time a word is removed from the buffer.
4	TIT	Offset relative to the beginning of the buffer which triggers an interrupt when the TOP pointer equals it.
5	TCM	Command to be loaded into the 2651 PCI when the transmit circular buffer is empty and the transmit shift register goes empty. The buffer is considered to be empty when the TOP pointer catches TIP. Bit-0 must be set for the command to be executed and bits 8-15 must be the command.

6-5. DATA OUT C (DOC) INSTRUCTION. The DOC instruction is used to specify the location of the receiver DMA control block in memory.

DOC	where AC3:	bits 9-15 = device code
	AC0:	address of receiver control block

An address of zero implies no receiver control block. This should be done when the device is closed.

This instruction must be executed to initialize DMA mode and when the entries in the DMA control block are changed by the CPU after the buffer has gotten full.

The DMA control block describes a circular buffer into which data received is placed by the controller. The buffer is defined by an initial buffer address (RBA), a length (RBL), an input pointer (RIP), and an output pointer (ROP). The RIP and ROP pointers are both relative to the origin of the buffer. The input pointer defines where the next data entry is to be put by the controller, and the output pointer defines where the next data entry is to be taken out by the CPU. The buffer is called circular because when either of the pointers RIP or ROP gets equal to RBL, it is reset to zero. Thus, the buffer contains data from ROP up to RIP if RIP > ROP, or from ROP up to RBLGTH and from the beginning up to RIP if RIP < ROP.

When DMA input is active, the controller is putting data into the buffer while advancing the RIP pointer, and the CPU is chasing along behind taking data out of the buffer while advancing the ROP pointer. The buffer is empty when ROP catches RIP. The buffer is full when RIP catches ROP-1. The buffer can be treated as a normal linear non-wraparound buffer by leaving ROP pointing at the beginning of the buffer.

A data entry in the receiver DMA buffer consists of the 2651 Status in bits 0-7 at the time the character was received and the received character right justified in bits 8-15.

No data can be received unless the receiver has been enabled by a command to the 2651.

The receiver control block contains the following parameters:

Word	Name	Meaning
0	RBA	Address in memory of receiver circular DMA buffer.
1	RBL	Length of buffer in words.
2	RIP	Offset relative to beginning of buffer where the next data/status received is to be put by the controller. This parameter is updated by the controller each time a data entry is stored in the buffer.
3	ROP	Offset relative to beginning of buffer where the CPU is to pick up the next word from the buffer. This word must be updated by the CPU each time a data word is removed from the buffer by the CPU.
4	RIT	Offset relative to the beginning of the buffer which causes an interrupt when the RIP pointer equals it.
5-9		Five words that contain characters which are to be matched with the received status/data. A word consisting of the 2651 status in bits 0-7 and the received character in bits 8-15 is anded with the mask X'28FF and compared with the words in the match block. If a match occurs, an interrupt is triggered. The two bits of the status are allowed to match so that in asynchronous mode it is possible to not match characters with parity or framing errors, and in synchronous transparent mode it is possible to match characters preceded by a DLE or SYN.

6-6. DATA IN A (DIA) INSTRUCTION. The DIA instruction is used to read one of the 2651 internal registers.

DIA	where AC3:	bit-0 = 1 means clear interrupt and inhibit further interrupts.
		bit-1 = 1 means clear interrupt and enable further interrupts.
		bits 9-15 = device code.

When the DIA instruction is executed, the controller will read the 2651 internal register specified by bits 6-7 of AC0 in the last DOA instruction executed. The data read from the 2651 will be stored in bits 8-15 of CPU register AC0. Controller and modem status flags will be stored in bits 0-7 of CPU register AC0. The controller/modem status bits are described below.

Bits	Meaning
0	Data Set Ready (DSR) from modem: 0=off, 1=on.
1	Delayed Transmitter Empty: 0=off, 1=on. This is the Transmitter Empty signal from the 2651 chip delayed by one transmitter clock time.
2	Data Carrier Detect (DCD) from modem: 0=off, 1=on.
3	Clear To Send (CTS) from modem: 0=off, 1=on.
4	Ring Indicator (RI) from modem: 0=off, 1=on. This signal is turned on when a ring is detected at the modem. It is kept on until a DIA instruction is executed that reads the 2651 status register.

The 2651 Command Register and Mode Registers are described earlier. The data previously loaded into these registers is simply given back when these registers are read. For the DIA instruction, the Receiver Holding Register replaces the Transmitter Holding Register and the Status Register replaces the SYN1/SYN2/DLE Registers.

Receiver Holding Register

The 2651 Receiver Holding Register contains the last character received. Bit-14 of AC0 will be set after reading the 2651 Status Register if the Receiver Holding Register contains a character. After the last bit of the character being received is assembled into the Receiver Shift Register and the stop bits have been checked in asynchronous mode, the character is copied to the Receiver Holding Register and an interrupt is triggered if the receiver DMA mode is not on. The program must then read this character from the Receiver Holding Register before the next received character has been completely assembled in the Shift Register or an overrun error will occur and the character will be lost. When the Receiver Holding Register is read, the character is returned in bits 8-15 of AC0. If the character length is less than 8, the character will be right justified and the most significant bits of the character field will be zero. If parity is enabled, the parity bit is stripped off. The Receiver Holding Register should be read directly only if the receiver DMA feature is not being used.

Status Register

The 2651 Status Register contains flags indicating the status of the 2651 chip and some modem signals. The bits in the 2651 Status Register have the following meaning as they appear in CPU register AC0.

Bits	Meaning
8	Data Set Ready (DSR) from modem: 0=off, 1=on
9	Generated Carrier Detect: 0=off, 1=on This signal is generated by the controller from the Data Carrier Detect (DCD) modem signal, and bit-2 of AC0 on the DOA command.
10	In asynchronous mode, 1 = character framing error. In synchronous mode, 1 = sync character(s) detected.
11	0=normal, 1=overrun error on receiver.
12	In asynchronous mode, 1 = parity error. In synchronous mode, 1 = parity error or DLE character detected.
13	Transmit Shift Register status: 0=full, 1=empty.
14	Receiver Holding Register status: 0=empty, 1=full
15	Transmit Holding Register status: 0=full, 1=empty

In receiver DMA mode, the Status Register error flags (bits 10, 11 and 12 described above) will be reset by the controller immediately after the character on which the error occurred has been transferred to the DMA buffer. In non-DMA mode, the program must reset the error flags.

6-7. DATA IN B (DIB) INSTRUCTION. The DIB instruction is used to read back the address of the transmitter DMA control block previously defined by a DOB instruction.

DIB where AC3 bits 9-15 = device code

6-8. DATA IN C (DIC) INSTRUCTION. The DIC instruction is used to read back the address of the receiver DMA control block address, previously defined by the DOC instruction.

DIC where AC3: bits 9-15 = device code

6-9. NO INPUT/OUTPUT (NIO) INSTRUCTION. The NIO instruction is used to RESET the 2651 chip and clear both the transmit and receiver DMA control block addresses. A reset of the 2651 chip causes all internal registers to be cleared. This turns off the Data Set Ready and Request to Send modem signals, puts the chip on external clocks for both the receiver and transmitter, and causes the chip to hang in SPACE state (0) on the Transmit Data line. It should normally not be necessary to use this instruction.

NIO where AC3: bits 9-15 = device code

6-10. MASK OUT (MSKO) INSTRUCTION. The MSKO instruction can be used to disable the interrupt logic for the communications controller. This is accomplished through the MSKO and a "1" in bit 8 of the word in AC0. The interrupt logic may be enabled through another MSKO and a "0" in bit 8.

6-11. INTERRUPT PROCESSING.

Program interrupts from the communications controller are cleared and further interrupts are disabled by a CLEAR function (reg AC3 bit-0 = 1) on a DOA or DIA instruction to the device.

Program interrupts are cleared and further interrupts are enabled by a START function (reg AC3 bit-1 = 1) on a DOA or DIA instruction to the device.

If both a START and CLEAR function are done at the same time, the result is the same as if only a CLEAR was done.

It is generally necessary for the macroprogram to read the 2651 Status register to determine why an interrupt occurred.

A summary of the events which trigger a macroprogram interrupt are given below. Since events happen on the communication line asynchronously with respect to the CPU, further interrupts may be triggered by events that happen while the macroprogram is already processing an interrupt.

1) If the receiver is enabled and not in DMA mode, an interrupt is triggered when the Receiver Holding Register goes full.

2) If the receiver is enabled and in DMA mode, an interrupt is triggered when the Receiver Holding Register goes full and the buffer is full, or after the controller stores a received character in the circular buffer and advances the input pointer (RIP) which then equals the interrupt trigger pointer (RIT), or if the status-character just received matches one of the entries in the match block, or if an overrun error was detected.

3) If the transmitter is enabled and not in DMA mode, an interrupt is triggered when the Transmitter Holding Register goes empty, or the Delayed Transmitter Empty signal changes state.

4) If the transmitter is enabled and in DMA mode, an interrupt is triggered when the Transmitter Holding Register goes empty and the buffer is empty, or after the controller fetches a character from the circular buffer and advances the output pointer (ROP) which then equals the interrupt trigger pointer (TIT) indicating the buffer is empty.

5) The modem signal Data Set Ready (DSR) goes on or off.

6) The modem signal Data Carrier Detect (DCD) goes off.

7) The modem signal Clear to Send (CTS) goes off.

8) The modem signal Ring Indicator (RI) goes on.

6-12. RBCC8 INSTRUCTION.

The RBCC8 instruction is used to compute block checksum characters for communication messages. This instruction is described in detail in Section 2.

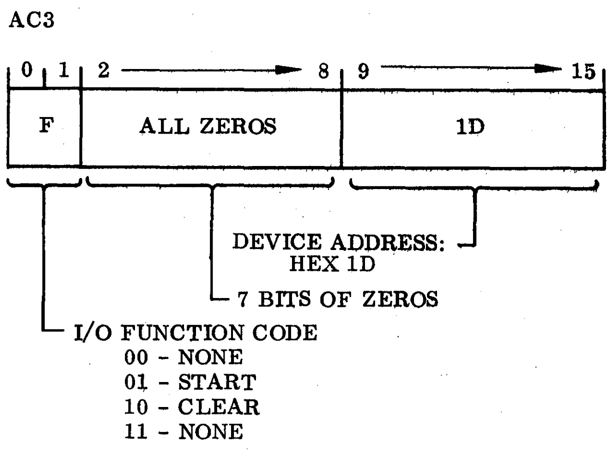
SECTION 7. AUTO-DIALER CONTROLLER

7-1. INTRODUCTION.

The Auto-Dialer Controller interfaces to an RS-366 compatible connector on the rear of the J500 which must be connected to a Bell 801 compatible auto dial unit. The processor communicates with the controller via Data Out A and Data In A instructions directed to a device address of Hex 1D.

7-2. I/O INSTRUCTIONS.

All I/O instructions require the simultaneous use of CPU accumulators AC0 and AC3. AC3 shown below contains the I/O address and a function code.

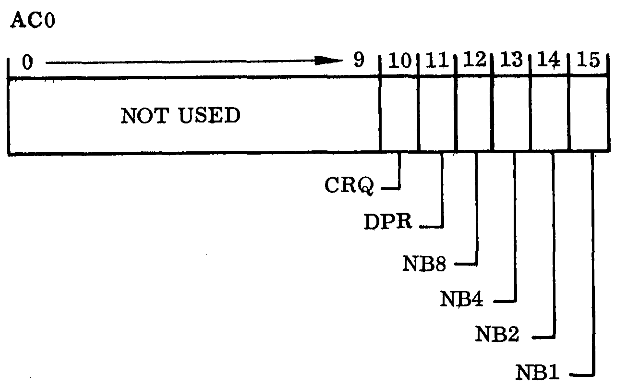


START - CLEAR CURRENT INTERRUPT AND ENABLE FURTHER INTERRUPTS.

CLEAR - CLEAR CURRENT INTERRUPT AND INHIBIT FURTHER INTERRUPTS.

7-3. DATA OUT A (DOA) INSTRUCTION.

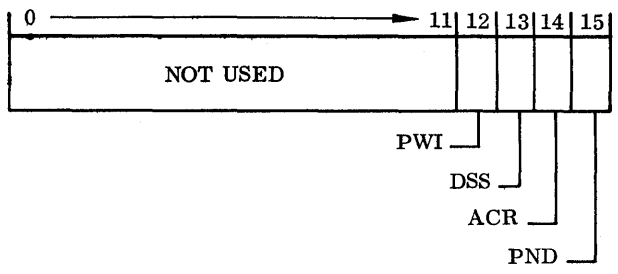
The DOA instruction is used to output digits and control flags to the Auto-Dialer. The format for data in AC0 is shown below:



Bit 10	CRQ - Call Request, 1 = start call
Bit 11	DPR - Digit Present, 1 = output digit
Bit 12	NB8 - Bit 8 of dial digit
Bit 13	NB4 - Bit 4 of dial digit
Bit 14	NB2 - Bit 2 of dial digit
Bit 15	NB1 - Bit 1 of dial digit

7-4. DATA IN A (DIA) INSTRUCTION.

The DIA instruction is used to read the status of the Auto-Dialer. The format of the data returned in AC0 is shown below:



Bit 12	PWI - Power Indicator, 1=power on
Bit 13	DSS - Data Set Status, 1=call answered and control transferred to Data Set
Bit 14	ACR - Abandon Call/Retry, 1=call abandoned
Bit 15	PND - Present Next Digit, 1=ready for new digit

7-5. MASK OUT (MSKO) INSTRUCTION. The MSKO instruction can be used to disable the interrupt logic for the Auto-Dialer. This is accomplished through the MSKO and a "1" in bit 8 of the word in AC0. The interrupt logic may be enabled through another MSKO and a "0" in bit 8.

7-6. INTERRUPT PROCESSING.

If interrupts are enabled, an interrupt is triggered on a change of any of the dialer status signals PWI, DSS, ACR, and PND. The macroprogram can determine the reason for the interrupt by reading the Auto-Dialer status.

7-7. PROGRAMMING RULES.

The control signal CRQ must be turned on to initiate the call. After a dial tone is received, the auto-dialer will turn on the status signal PND indicating that it is ready for the computer to present a new digit. If the phone line is occupied (off-hook) when CRQ is turned on, the dialer will not turn on PND, but instead will time out and turn on the abandon call/retry status signal ACR.

The computer must respond to the PND signal by outputting a new digit on the data lines NB1-NB4 with the DPR control flag turned on to indicate that a new digit is present on the interface. The digits are represented by a 4-bit binary number.

When the digit has been accepted by the dialer, it will turn off the PND status signal.

The computer must then respond by turning off the DPR control line.

When the dialer is ready for another digit, it will turn on the PND line. The sequence for outputting digits described above is repeated until all of the digits have been output.

After the last digit has been output, the dialer will again turn on PND. This time the computer does not respond.

When an answer tone is received on the phone line, the dialer will transfer control of the phone line to the data set to which the dialer is connected. The dialer status signal DSS will be turned on when transfer of control is complete.

If the dialer receives a busy signal after any digit, or does not receive an answer tone within a preset amount of time after the last digit, it will turn on the ACR status signal indicating that the call should be abandoned. The computer should then turn off the CRQ control line for at least 15 seconds before attempting to retry the call.

Under some conditions, the dialer must wait for another dial tone after some of the digits have been output. This happens when dialing through a local PBX and is known as "tandem dialing". Not all auto-dial units are capable of doing this. In those that can, the request for a new dial tone is usually indicated by outputting an invalid digit (10, 11, 13, 14, or 15).

On some auto-dialers, a 12 digit indicates that control is to be transferred to the associated data set before an answer tone is detected.

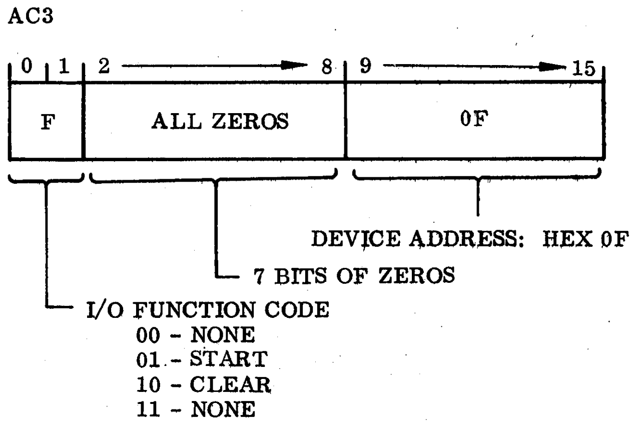
The auto-dial units, a 12 digit indicates that control new digit is not output within 5 minutes of the previous one, the CRQ control line will be reset causing the call to be abandoned. This is to prevent the computer from tying up the phone line for a long time in the case where the system crashes or the program using the dialer malfunctions.

8-1. INTRODUCTION.

The J500 Printer Controller provides control of one printer of either character or line type. Addressing and control of the two types of printers are described in the following paragraphs.

8-2. LINE PRINTER I/O INSTRUCTIONS.

The J500 will interface to selected printers with Data Products or Centronics type interfaces. All I/O instructions require the simultaneous use of CPU accumulators AC0 and AC3. AC3 shown below contains the I/O address and a function code. A device address of Hex 0F must be used to address a line printer.



8-3. DATA OUT A (DOA) INSTRUCTION.

The DOA instruction is used to transfer one character of data to the printer. The data is always contained in bits 8 through 15 for Centronics printers and 9 through 15 for Data Products printers (see Figure 8-1). The data characters and codes are listed in Table 8-1. Characters LF, CR, and FF (Table 8-1) are the three characters available to the user for format control. The three format characters perform the following:

1. LF (Line Feed). The LF command advances the paper one line and clears the column register, returning it to the leftmost print position. If the first character in the new set is another LF instruction, the paper continues slewing for another line. Slewing will continue, one line for every LF command as long as LF commands are transmitted. A top-of-form cam encountered during a given paper feed will cause the paper to slew for 3 or 4 lines depending on timing belt placement and thus skip over perforations.

2. CR (Carriage Return). The CR command clears the column register and initializes the zone control register.

3. FF (Form Feed). The FF command advances the paper to the top of the next form.

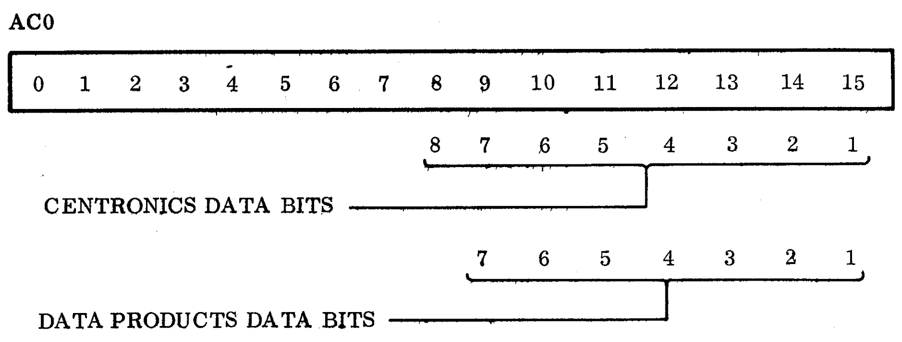
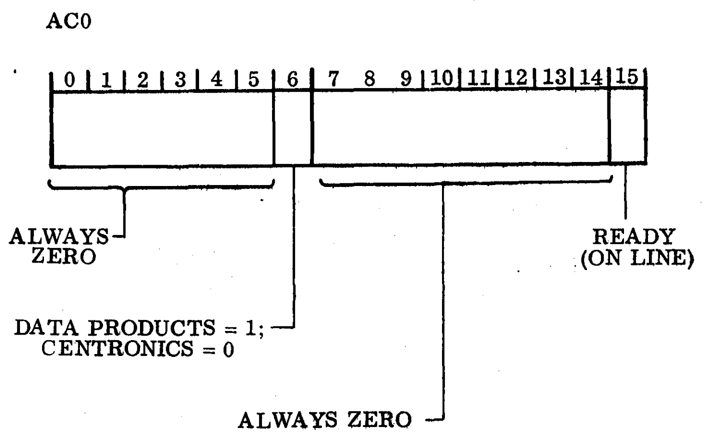


Figure 8-1. Line Printer Data Character - DOA Instruction

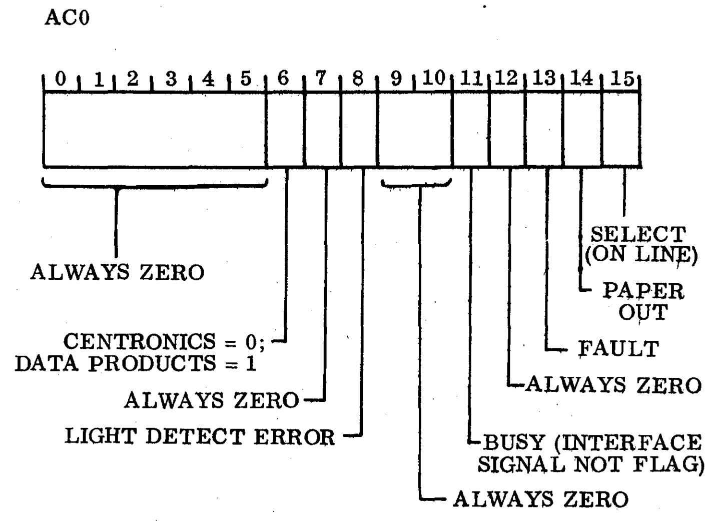
Table 8-1. Line Printer Character Set

b7 b6 b5				0 0 0	0 1 0	0 1 1	1 0 0	1 0 1
b4	b3	b2	b1					
0	0	0	0		Space	Ø	@	P
0	0	0	1		!	1	A	Q
0	0	1	0		"	2	B	R
0	0	1	1		#	3	C	S
0	1	0	0		$	4	D	T
0	1	0	1		%	5	E	U
0	1	1	0		&	6	F	V
0	1	1	1		'	7	G	W
1	0	0	0		(	8	H	X
1	0	0	1		)	9	I	Y
1	0	1	0	LF	*	:	J	Z
1	0	1	1		+	;	K	[
1	1	0	0	FF	,	<	L	◇
1	1	0	1	CR	-	=	M	]
1	1	1	0		.	>	N	∧
1	1	1	1		/	?	O	♡

8-4. DATA IN A (DIA) INSTRUCTION. The DIA instruction is used to read the status of the printer into AC0. The format for Data Products type and Centronics type printers is shown below:



DATA PRODUCTS FORMAT



CENTRONICS FORMAT

Figure 8-2. Line Printer DIA Format

8-5. MASK OUT (MSKO) INSTRUCTION. The MSKO instruction can be used to disable the interrupt logic for the printer. This is accomplished through the MSKO and a "1" in bit 12 of AC0. The interrupt logic may be enabled through another MSKO and a "0" in bit 12.

8-6. CHARACTER PRINTER I/O INSTRUCTIONS.

The J500 will interface with Diablo, Qume or NEC type character printers. All I/O instructions require the simultaneous use of CPU accumulators AC0 and AC3 as described in Paragraph 8-2. The character printers use a Device Select code of Hex 0E for I/O operations.

8-7. DATA OUT A (DOA) INSTRUCTION. The DOA instruction is used to transfer a command plus control information or an ASCII character. The format for the Data Out A is shown in Figure 8-3. The format for the various commands is shown in Figures 8-4 through 8-8. The ASCII characters and their meaning are listed in Tables 8-2 and 8-3.

8-8. DATA IN A (DIA) INSTRUCTION. The DIA instruction is used to get printer status information. The word format and status information is shown in Figure 8-9.

8-9. MASK OUT (MSKO) INSTRUCTION. The MSKO instruction is the same as for the line printer described in Paragraph 8-5.

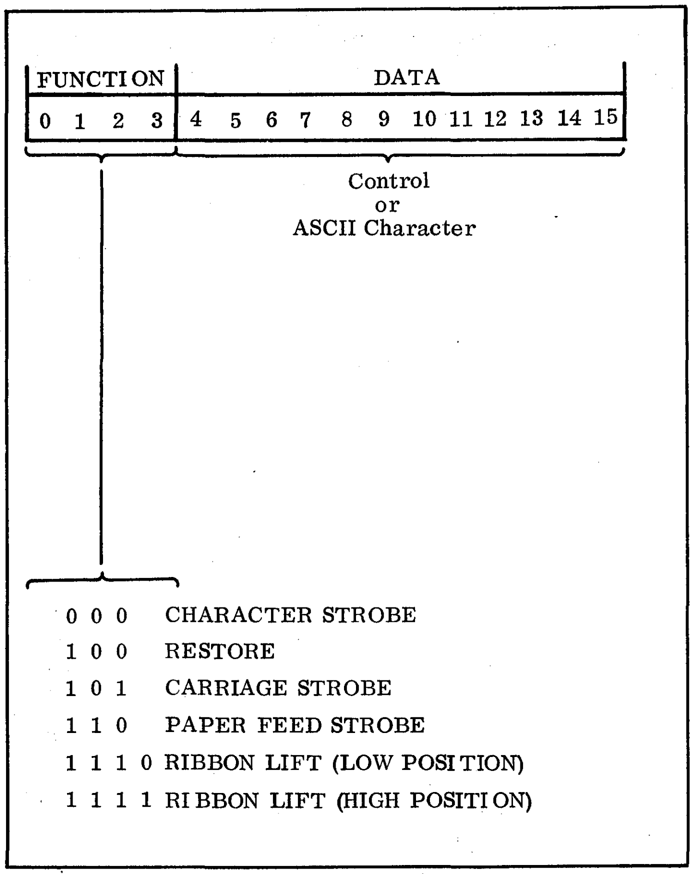


Figure 8-3. DOA Word Format

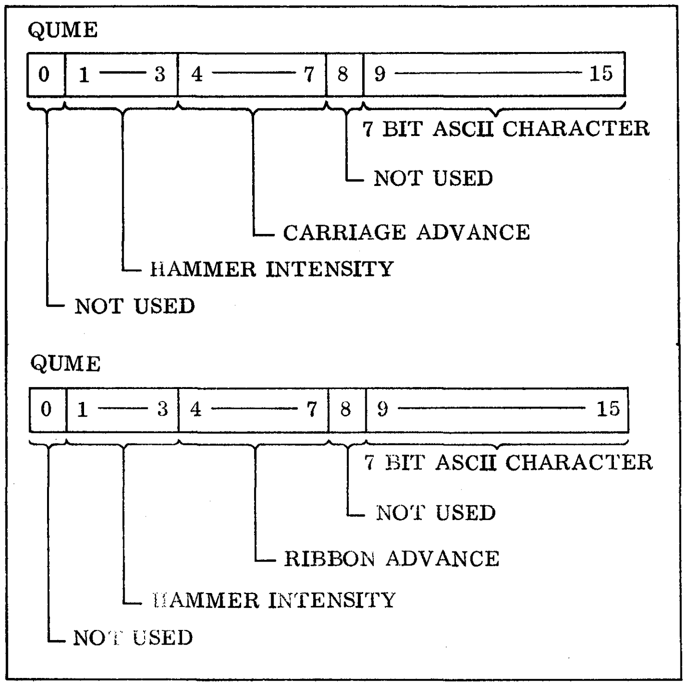


Figure 8-4. Print Command Word Formats

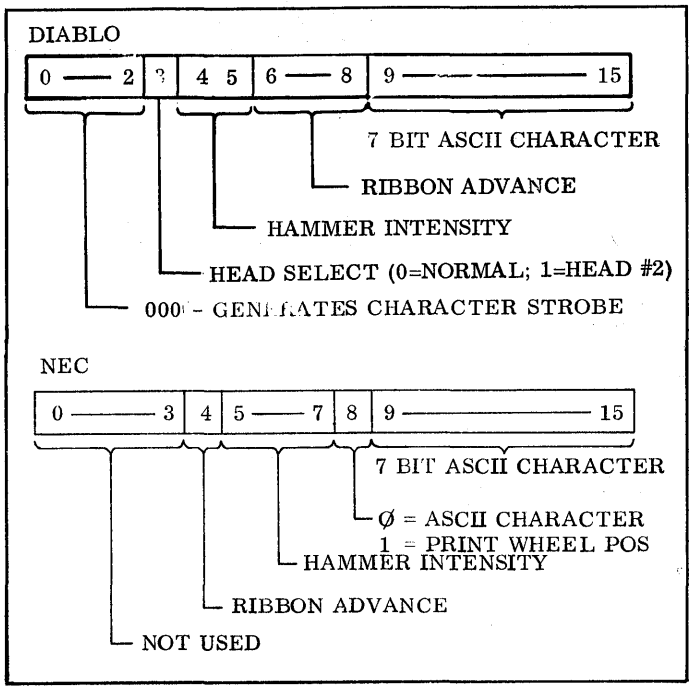


Figure 8-4. Print Command Word Formats (Cont)

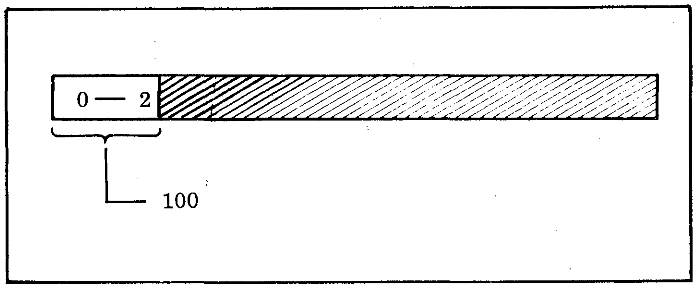
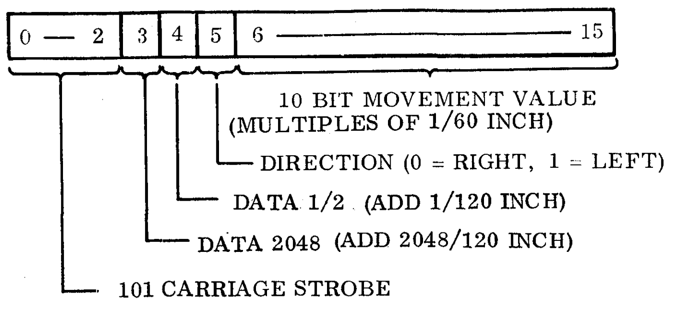


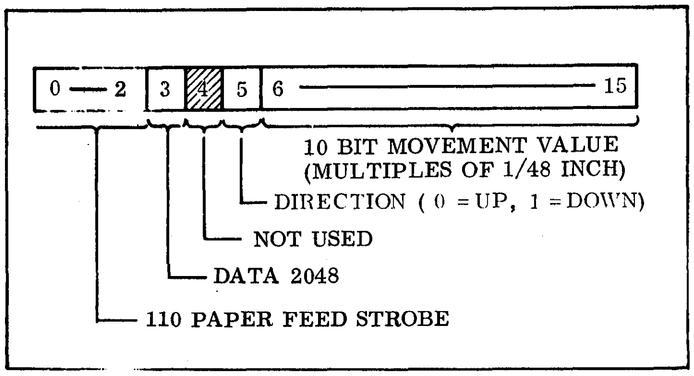
Figure 8-5. Restore Command


0 — 2 | 3 | 4 | 5 | 6 — 15
10 BIT MOVEMENT VALUE (MULTIPLES OF 1/60 INCH)
DIRECTION (0 = RIGHT, 1 = LEFT)
DATA 1/2 (ADD 1/120 INCH)
DATA 2048 (ADD 2048/120 INCH)
101 CARRIAGE STROBE

NOTE

Character spacing for non-proportional printers is 10 characters per inch; thus, to output a "Space" code, the carriage movement command requires a value of six (6) in the lower ten bit field. Also, a space is required after each Print Command to move the carriage to the next character position. The number of characters per inch for proportional spacing printers will vary depending on the characters that are printed.

Figure 8-6. Carriage Strobe Command



0 — 2
3
4
5
6 — 15
10 BIT MOVEMENT VALUE (MULTIPLES OF 1/48 INCH)
DIRECTION (0 = UP, 1 = DOWN)
NOT USED
DATA 2048
110 PAPER FEED STROBE

Figure 8-7. Paper Feed Command

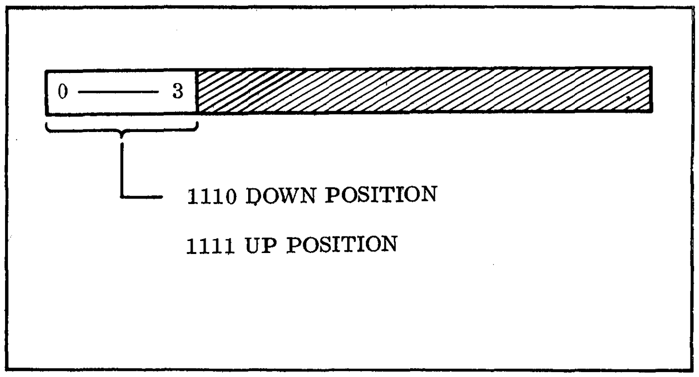
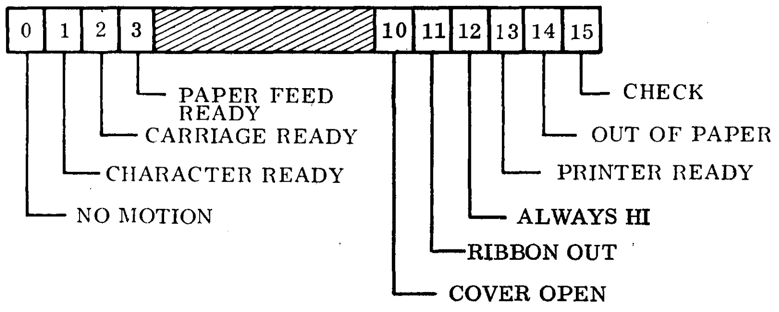


Figure 8-8. Ribbon Lift Command



Bit 0	NO MOTION	– An "AND" condition of bits 1, 2 and 3 indicates the printer is ready for another command.
Bit 1	CHARACTER READY	– Indicates the printer is ready for another character command.
Bit 2	CARRIAGE READY	– Indicates the printer is ready for a new carriage command.
Bit 3	PAPER FEED READY	– Indicates the printer is ready for a new paper feed command.
Bit 10		– Indicates printer cover is open. Printer will not accept a new command.
Bit 11		– Indicates printer is out of ribbon.
Bit 12	ALWAYS HI	– Always Hi
Bit 13	PRINTER READY	– Indicates that the printer power is on.
Bit 14	OUT OF PAPER*	– Indicates the printer is out of paper.
Bit 15	CHECK	– Indicates machine malfunction. Bits 0, 1, 2 and 3 will also be false. Printer must be reset by a Restore command or manually by switching Printer Power OFF then ON.

*NOTE: If a paper sensor is not installed, this line will always indicate "out of paper."

Figure 8-9. Status Word Format

Table 8-2. Seven Bit ASCII Code for 96 Character Diablo HyType II and Qume Printers

			(MSB) b7	0	0	1	1	1	1
			b6	1	1	0	0	1	1
b4	b3	b2	b1 b5	0	1	0	1	0	1
0	0	0	0	¢	0	@	P	SEE NOTE 1	p
0	0	0	1	!	1	A	Q	a	q
0	0	1	0	″	2	B	R	b	r
0	0	1	1	#	3	C	S	c	s
0	1	0	0	$	4	D	T	d	t
0	1	0	1	%	5	E	U	e	u
0	1	1	0	&	6	F	V	f	v
0	1	1	1		7	G	W	g	w
1	0	0	0	(	8	H	X	h	x
1	0	0	1	)	9	I	Y	i	y
1	0	1	0	*	:	J	Z	j	z
1	0	1	1	+	;	K	[	k	{
1	1	0	0	,	<	L	\	l	|
1	1	0	1	-	=	M	]	m	}
1	1	1	0	.	>	N	∧	n	~
1	1	1	1	/	?	O	—	o	¬

NOTES 1. Blank on Diablo wheel; Grave accent (\) symbol on Qume wheel.

Table 8-3. Seven Bit ASCII Code for 88 Character Diablo Word Processing Printer

			(MSB) b7	0	0	1	1	1	1
			b6	1	1	0	0	1	1
b4	b3	b2	b1 b5	0	1	0	1	0	1
0	0	0	0	¢	0	@	P	′	p
0	0	0	1	!	1	A	Q	a	q
0	0	1	0	″	2	B	R	b	r
0	0	1	1	#	3	C	S	c	s
0	1	0	0	$	4	D	T	d	t
0	1	0	1	%	5	E	U	e	u
0	1	1	0	&	6	F	V	f	v
0	1	1	1	'	7	G	W	g	w
1	0	0	0	(	8	H	X	h	x
1	0	0	1	)	9	I	Y	i	y
1	0	1	0	*	:	J	Z	j	z
1	0	1	1	+	;	K		k	•
1	1	0	0	,	¼	L		l	
1	1	0	1	-	=	M		m	
1	1	1	0	.	½	N		n	
1	1	1	1	/	?	O	—	o	

9-1. INTRODUCTION

This section contains information on initializing the J500 and information that may appear on the CRT as error or status messages. During the power-up sequence, the J500 initialize routine checks logic, registers, buses and memory for possible errors prior to loading the operating program from the selected device.

9-2. APL SWITCHES

A rocker switch on the lower right side of the J500 allows selection of either a cartridge drive or diskette drive as the device from which the APL will boot. This switch is used in conjunction with one switch in the DIP switch package in location 24H on the multiwire logic board to enable selection of a floppy drive or cartridge drive as the input device from which the CPU will load the Operating System.

Switches 5, 6, and 7 of the DIP switch package are used to select the APL device. If all switches are in the OFF position, the CARTRIDGE/DISKETTE switch can be used to select either FP00 or DD00. Other switches in the DIP switch package perform other functions as described in Table 9-1.

CAUTION

If the APL or other switches must be changed, the equipment must be turned off and the power cable disconnected before the J500 cover is removed to change the switches.

Table 9-1. DIP Switch Functions - Location 24H

Switch	Function
SW1	Floppy Fast Step. When ON, the controller will generate Fast Step pulses for the floppy drives. For drives without the capability of Fast Stepping, a noise will be heard coming from the drive because it cannot handle the Fast Step pulses. This switch is set at the factory and should not be changed.
SW2	50 CPS. ON is for 50 Hz operation and OFF is for 60 Hz operation.
SW3	Loop. When ON, the processor will loop in the initialize routine from the beginning up to the point where the APL switches are tested and then back to the beginning where the routine is repeated.
SW4	Not Used.
SW5	Floppy Unit APL Select. When ON, will boot from FLOPPY 1 if Floppies are the selected APL source.
SW6	Cartridge Unit APL Select. When ON, will boot from the fixed platter of the cartridge if the cartridge is the selected APL source.
SW7	Floppy APL Source. When ON, overrides outer Boot Select Switch and forces a boot from Floppy.
SW8	Enable Extended Memory Test. When OFF, the initialize routine will include an extended memory test that takes approximately 6 seconds. When ON, the test will be skipped.

9-3. INITIALIZE PROCEDURE

This paragraph describes the procedure for initializing the J500. Events that take place during the initialize routine and messages that may appear on the screen are described in Paragraph 9-4.

CAUTION

Do not turn power on or off to the J500 while diskettes are loaded or while the cartridge drives are READY.

1. Ensure that all connectors to the J500 are properly connected and mated.
2. Set POWER switch to ON.
3. Turn ENABLE/DISABLE keyswitch clockwise (DISABLE) to initialize system.
4. Set CARTRIDGE/DISKETTE switch to device which APL will boot from. Push top in for CARTRIDGE or bottom in for DISKETTE.
5. Turn ENABLE/DISABLE keyswitch counterclockwise to enable system.

NOTE

A blinking "ERROR" message will appear on the top of the screen if an error is detected during the initialize routine.

6. Load the floppy diskette or turn the cartridge drive on to boot from the appropriate device.

The processor will boot from the specified device and if the boot is successful, the word "BOOT" will appear on the bottom of the screen.

If the boot is unsuccessful, the word "STATUS" will appear on the top of the screen followed by a status code. The status code is the status word for the Universal Disk Controller as described in the Data In A instruction for that controller.

9-4. INITIALIZE ROUTINE

The J500 automatically goes through an initialize routine each time the ENABLE/DISABLE keyswitch is turned from DISABLE to ENABLE. If there are no errors during the initialize routine, the CRT will display a string of 128 characters followed by a string of 128 blanks for the 128 programmable characters. Figure 9-1 illustrates a normal CRT display before Auto Program Load.

If an error is detected. a blinking "ERROR" message appears on the top line of the screen followed by four rows of four hexadecimal digits in lines 2-5 columns 1-4. The four rows of digits will be 0's unless a memory error was detected in which case a "1" will be inserted into the error message to correspond to the row and chip number where the error was detected. An example of an error message is shown below:

ERROR	NOTES
0000	A Row; Addresses C000 - FFFF
0020	B Row; Addresses 8000 - BFFF
0000	C Row; Addresses 4000 - 7FFF
0000	D Row; Addresses 0000 - 3FFF

The above message indicates an error was detected in the 6th chip from the right in Row B.

After the basic initialize routine has been completed, the APL switches are tested and registers R0-R3 are loaded with information and appropriate values for that device. The register contents and values are shown below:

R0 = Head/Platter Number

R1 = Sector Number

R2 = 0000

R3 = Device Code

	R0	R1	R2	R3
FP00	0000	0001	0000	0014
FP01	0000	0001	0000	0015
DD00	0000	0000	0000	0018
DD01	0002	0000	0000	0018

The APL program is loaded into memory starting at address '7B80'. The APL program executed at address '7B80' will do the following:

1. Test memory location '0000' where errors are reported. If an error was reported, the top four locations of the stack are converted into ASCII and displayed in lines 2-5 columns 1-4 below the word "ERROR".
2. Restore FP00. Wait until not busy or timed out.
3. Restore FP01. Wait until not busy or timed out.
4. If booting from cartridge, skips the next step.
5. Restore and read from selected device with the double density option. Wait until not busy or timed out. Display the status on the top line as "D-STATUS XXXX" where XXXX is the disk controller status. If status is good, go to step 7.
6. Restore and read from selected device without the double density option. Wait until not busy or timed out. Display the status and blank out the "D-" in the message.

   If status is good, go to step 7.

   If status is bad, return to beginning of initialize routine.
7. Set up registers R2 and R3 as shown below, then jump to address '0001'.

   R2 = Address of Disk Control Block.

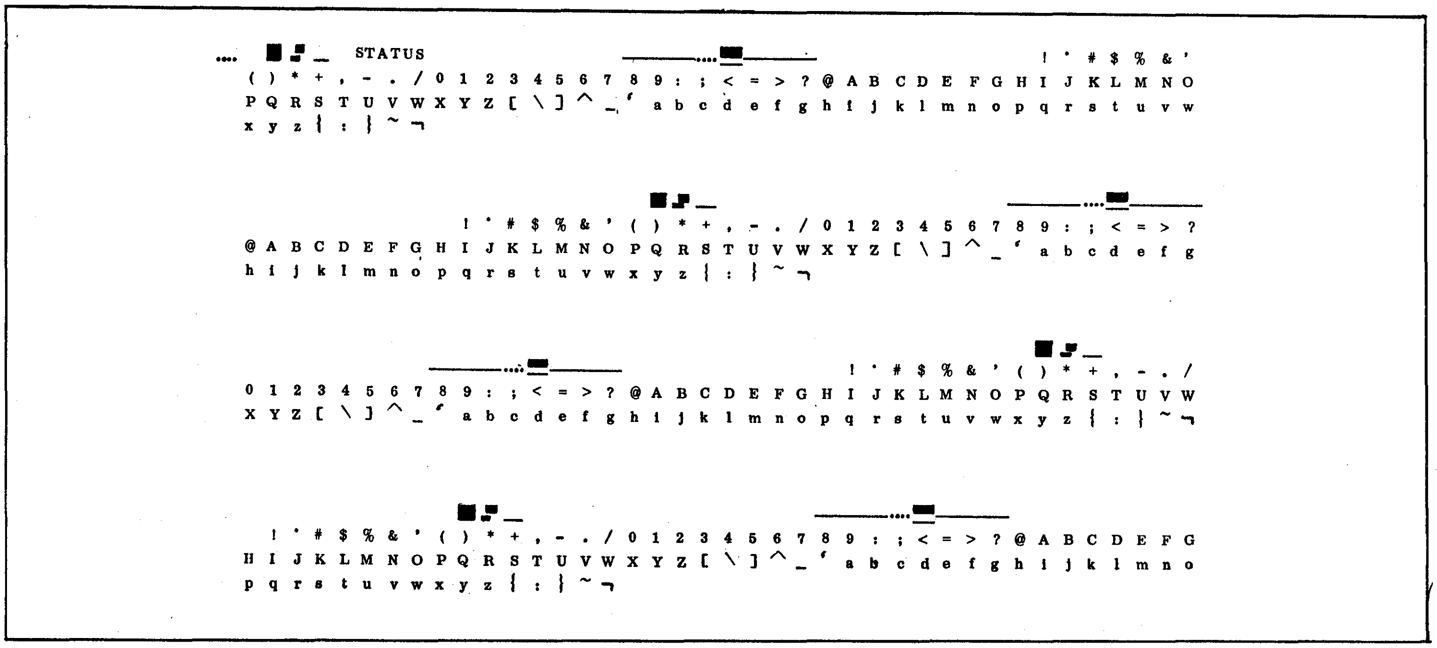
   R3 = Disk Device Code

9-5. LOOPING IN INITIALIZE ROUTINE

The processor can be made to loop in the initialize routine by setting a switch on the logic board (see Table 9-1) or by depressing the REPEAT key on the keyboard. The REPEAT key should be held depressed for approximately 6 seconds to make sure the Repeat function is detected in the initialize routine.

If the REPEAT key is held depressed, the loop will be repeated each time the processor senses the Repeat function and will continue until the REPEAT key is released.

To loop continuously in the initialize routine, depress the REPEAT key and then the character "T". Hold both keys depressed for approximately 6 seconds to make sure the continuous loop function has been detected. To exit from the loop, depress any character other than the character "T". If an error is detected while in the continuous loop, the speaker will continuously "BEEP" to indicate an error is detected.



```
....  ■ ▟ _  STATUS                    ────────....▬────────                 ! " # $ % & '
  ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H I J K L M N O
  P Q R S T U V W X Y Z [ \ ] ^ _ ` a b c d e f g h i j k l m n o p q r s t u v w
  x y z { : } ~ ¬


                                            ■ ▟ _                ────────....▬────────
                        ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ?
  @ A B C D E F G H I J K L M N O P Q R S T U V W X Y Z [ \ ] ^ _ ` a b c d e f g
  h i j k l m n o p q r s t u v w x y z { : } ~ ¬


                                                                         ■ ▟ _
                  ────────....▬────────                     ! " # $ % & ' ( ) * + , - . /
  0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G H I J K L M N O P Q R S T U V W
  X Y Z [ \ ] ^ _ ` a b c d e f g h i j k l m n o p q r s t u v w x y z { : } ~ ¬


                          ■ ▟ _                          ────────....▬────────
    ! " # $ % & ' ( ) * + , - . / 0 1 2 3 4 5 6 7 8 9 : ; < = > ? @ A B C D E F G
  H I J K L M N O P Q R S T U V W X Y Z [ \ ] ^ _ ` a b c d e f g h i j k l m n o
  p q r s t u v w x y z { : } ~ ¬
```

Figure 9-1. Normal Initialize CRT Display

APPENDIX A

SUMMARY OF INSTRUCTIONS

Table A-1. Instruction Set

INSTRUCTION	MNEMONIC	OP CODE BASE	EXECUTION TIME-USEC	MEMORY READ CYCLES	MEMORY WRITE CYCLES
Load and Store Instructions					
Load	LD	8000	2.64	2	-
Load Indirect*	LD@	9000	3.96	3	-
Store	ST	A000	2.64	1	1
Store Indirect*	ST@	B000	3.96	2	1
Arithmetic Instructions					
Add	ADD	C000	5.28^1	2	-
Subtract	SUB	D000	5.28^1	2	-
Logical Instructions					
And	AND	6000	2.64	2	-
Or	OR	6800	2.64	2	-
Skip Instructions					
Increment and Skip if Zero	ISZ	7800	3.96^2	2	1
Decrement and Skip if Zero	DSZ	7C00	3.96^2	2	1
Skip if Greater	SKG	E000	$3.96^{1,2}$	2	-
Skip if Not Equal	SKNE	F000	$3.96^{1,2}$	2	-
Skip if And is Zero	SKAZ	7000	$3.96^{1,2}$	2	-
Transfer-of-Control Instructions					
Jump	JMP	2000	3.96^1	1	-
Jump Indirect*	JMP@	2400	5.28^1	2	-
Jump to Subroutine	JSR	2800	5.28^4	1	-
Jump to Subroutine Indirect*	JSR@	2C00	6.60^4	2	-
Branch On Condition	BOC	1000	2.64^2	1	-
Return From Interrupt	RTI	0100	3.96	1	-
Return From Subroutine	RTS	0200	3.96	1	-
Shift Instructions					
Rotate Left	ROL	5800	$7.92 + (2.64N)^{5,6}$	1	-
Rotate Right	ROR	5800	$7.92 + (2.64N)^{5,6}$	1	-
Shift Left	SHL	5C00	$7.92 + (2.64N)^{5,6}$	1	-
Shift Right	SHR	5C00	$7.92 + (2.64N)^{5,6}$	1	-

* - The symbol @ must precede the designation of the memory location whose contents become the effective address by indirection.

Table A-1. Instruction Set (Continued)

INSTRUCTION	MNEMONIC	OP CODE BASE	EXECUTION TIME-USEC	MEMORY READ CYCLES	MEMORY WRITE CYCLES
Register Instructions					
Push on to Stack	PUSH	4000	5.28[3,4]	1	-
Pull from Stack	PULL	4400	2.64	1	-
Exchange Register and Stack	XCHSR	5400	3.96	1	-
Load Immediate	LI	4600	1.32	1	-
Add Immediate Skip if Zero	AISZ	4800	3.96	1	-
Complement and Add Immediate	CAI	5000	2.64	1	-
Register Add	RADD	3000	3.96	1	-
Register Exclusive Or	RXOR	3082	2.64	1	-
Register And	RAND	3083	2.64	1	-
Register Exchange	RXCH	3080	2.64	1	-
Register Copy	RCPY	3081	2.64	1	-
Compute Block Check Character	RBCC8	3084	6.60	1	-
Input Output Instructions					
Data In A	DIA	0401	3.96 + Δ	1	-
Data In B	DIB	0402	3.96 + Δ	1	-
Data In C	DIC	0404	3.96 + Δ	1	-
Interrupt Acknowledge	INTA	0410	3.96 + Δ	1	-
I/O Reset	IORST	0420	3.96 + Δ	1	-
No Data Transfer	NIO	0600	3.96 + Δ	1	-
Data Out A	DOA	0601	3.96 + Δ	1	-
Data Out B	DOB	0602	3.96 + Δ	1	-
Data Out C	DOC	0604	3.96 + Δ	1	-
Mask Out	MSKO	0608	3.96 + Δ	1	-
Read Done	RDNE	0620	3.96 + Δ	1	-
Read Busy	RBSY	0640	3.96 + Δ	1	-
Halt and Flag Instructions					
Pull Status Flags from Stack	PULLF	0280	3.96	1	-
Set Flag	SFLG	0800	1.32	1	-
Halt	HLT	0000	5.28	1	-
Push Status Flags on Stack	PUSHF	0080	6.60[3,4]	-	-
Pulse Flag	PFLG	0880	1.32	1	-
Interrupt Instructions					
Interrupt Enable	INTEN	0900	1.32	1	-
Interrupt Disable	INTDS	0980	1.32	1	-

Notes:

1. -1.32μs if Base Page Mode
2. $+1.32\mu$s if Skip
3. -2.64μs if Interrupt Enable = 0
4. $+1.32\mu$s if Interrupt Enable = 1 & Stack Full
5. 2.64μs if SEL = 0 & N = 8
6. N = Number of Shifts

JPS500-01

COMMENT SHEET

Jacquard Systems

FROM

Name ______________________________

Business Address ______________________________

____

____

Does this publication meet your requirements? yes ☐ no ☐

If not, please explain. ______________________________

____

____

____

Do you wish a reply? yes ☐ no ☐

COMMENTS

Describe any errors and/or suggested changes. Please include page number.

____

____

____

____

____

____

____

____

____

____

CUT ALONG DOTTED LINE

(FOLD)

FIRST
CLASS
PERMIT NO.
2609
SANTA MONICA
CA.

BUSINESS REPLY MAIL

NO POSTAGE STAMP NECESSARY IF MAILED IN UNITED STATES

POSTAGE WILL BE PAID BY:

Jacquard Systems

3340 Ocean Park Boulevard
Santa Monica, California 90405

Customer Service Department

(FOLD)

CUT ALONG DOTTED LINE

3340 Ocean Park Boulevard
Santa Monica, California 90405

Jacquard Systems

3340 Ocean Park Boulevard
Santa Monica, California 90405

Jacquard Systems