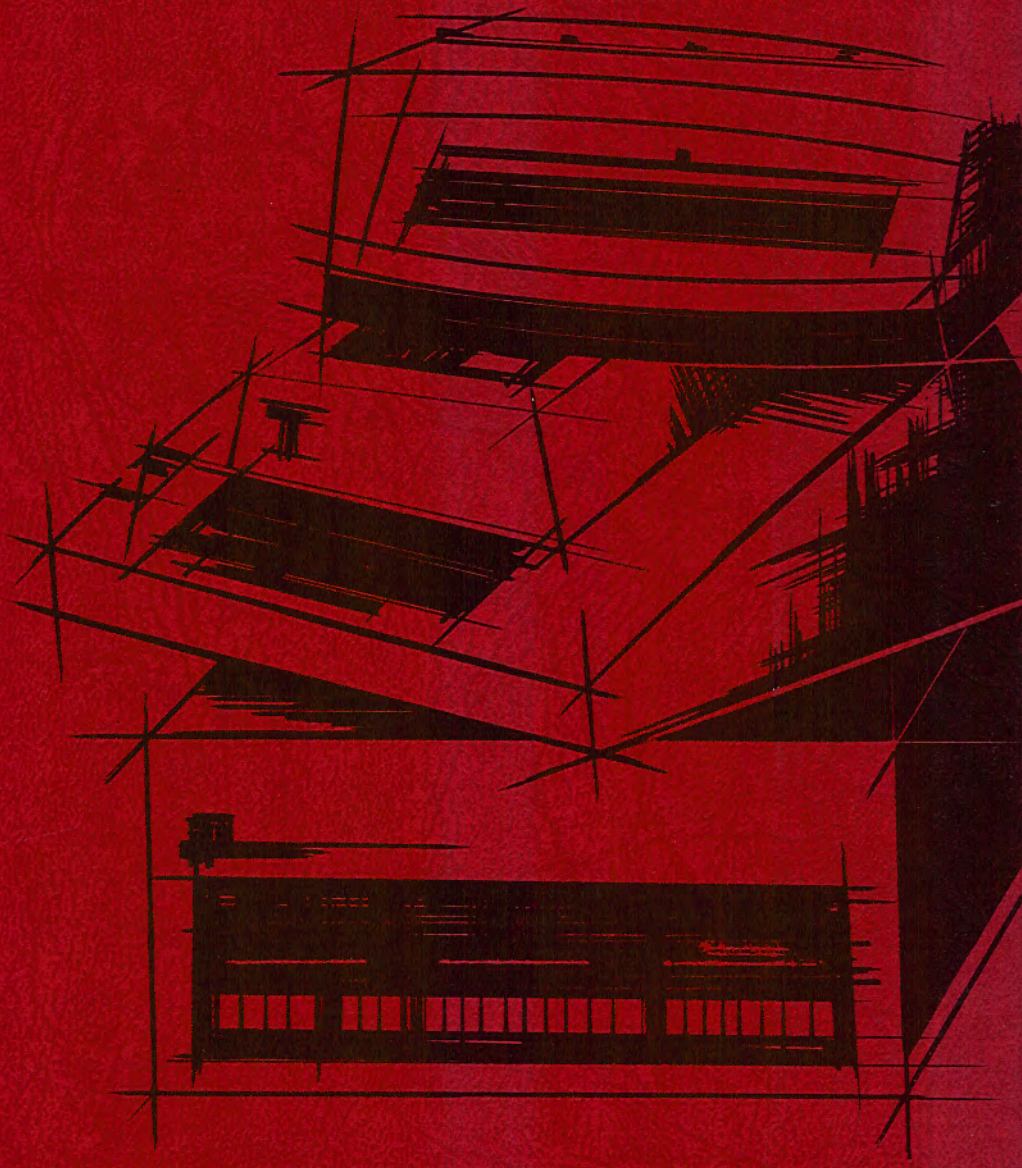
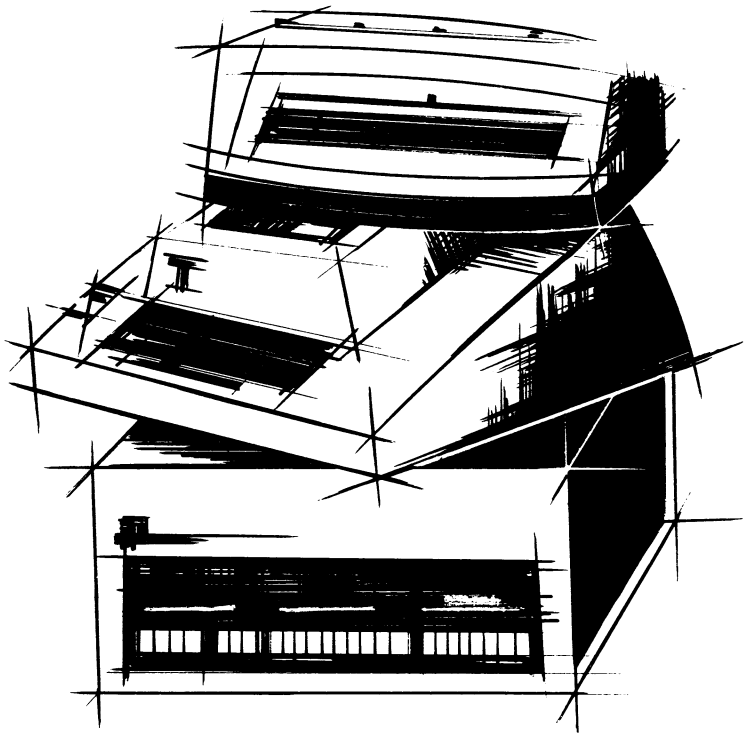
3300
COMPUTER REFERENCE MANUAL



WANG
LABORATORIES, INC.

3300

COMPUTER REFERENCE MANUAL



WANG
LABORATORIES, INC.

Tewksbury, Mass. 01876
Telephone (617) 851-7311
TWX 710 343-6769

TABLE OF CONTENTS

TABLE OF CONTENTS (Concl'd)

LIST OF ILLUSTRATIONS

LIST OF TABLES

I. INTRODUCTION

The Wang 3300 is an integrated circuit general purpose mini computer. Unlike most computers in this class, it was specifically designed to be the central processor for multi-terminal time-sharing applications for higher level language systems. The Wang 3300 design incorporates such features as:

- A powerful repertoire of 68 instructions, including 21 memory reference instructions.
- A flexible and powerful I/O logic design which includes an I/O bus structure to handle up to 128 low and medium speed peripheral devices and Direct Memory Access Channel logic to control high speed devices at transfer rates up to 300,000 cps.
- A highly efficient priority interrupt structure which handles up to 128 priority interrupt levels and incorporates device address interrupt acknowledgement to reduce interrupt processing time.
- Memory expandability up to 65k bytes to accommodate large system applications.
- Nine modes of addressing, including a unique auto-increment/auto-decrement indirect addressing mode throughout all of memory. In this mode the indirect address is decremented before and incremented after single or double byte memory transfers. This produces a true push-pop access mode for handling table and list processing efficiently. In many applications, it provides the flexibility and speed equivalent to an unlimited number of index registers, and also eliminates the time consuming bookkeeping requirements associated with index registers.
- Six arithmetic instructions which operate in both binary and decimal mode. In decimal mode, 4 bit decimal groups are operated on in decimal arithmetic. This feature eliminates the inaccuracies and time inefficiencies associated with decimal to binary conversion and binary arithmetic, and provides a multi-precision capability.

The Wang 3300 is a byte-oriented computer, but it also has a number of double byte operand memory reference commands. This in essence provides the best features of both 8 bit and 16 bit computers. In many areas where the processing involves higher level languages, a single byte or character is commonly referenced. A 16 bit fetch is inefficient and often involves additional masking logic. However, in those instances in which 16 bit transfers are required, Wang 3300 double byte commands accommodate them.

The Wang 3300 is designed specifically to meet the high performance requirements of time-sharing systems. However, a modular design and a flexible and powerful I/O structure and instruction repertoire enable the 3300 to meet a great variety of scientific and commercial applications in the both real-time and batch processing modes.

II. GENERAL SPECIFICATIONS

- Computer transfer logic - 8 bit parallel binary
- Memory Cycle time - 1.6 μ sec.
- Word Length - 8 bits
- Instruction Length - 16 bits
- Addressing - Memory is segmented in 256 byte pages.

  The following addressing modes are available.
  - Direct Absolute page (Page 0 or 1)
  - Direct Current page
  - Indirect Absolute page (Page 0 or 1)
  - Indirect Current page
  - Indirect, Absolute Page, Auto-Increment (Page 0 or 1)
  - Indirect, Absolute Page, Auto-Decrement (Page 0 or 1)
  - Indirect, Current Page, Auto-Increment
  - Indirect, Current Page, Auto-Decrement
  - Immediate
- Memory Type - Magnetic core
- Memory Size - From 4,096 to 65,536 bytes in increments of 4,096 bytes
- Arithmetic Modes - Binary Mode and Addressing - 2's complement
  Decimal Mode - 4 bit BCD
  Complement Instructions - 1's complement or 9's complement
- Speed
  Add: 8 bit binary - 4.8 μsec
  Add: 8 bit decimal - 4.8 μsec
  Add: 16 bit binary - 6.4 μsec
  Add: 16 bit decimal - 6.4 μsec
  Multiply: floating point, (8 digit decimal mantissa) 4 ms
- Input Output
  - I/O Bus structure which controls up to 128 devices
  - Multiplex Mode (character Buffers)
  - Direct Memory Access channel Mode
- Interrupt
  - Device Station Groups
  - 128 Priority Levels
- Peripheral Equipment

  Wang 3310 I/O Writer - (Modified IBM Selectric)
  - Keyboard Input
  - Typed Output at 15 cps

  Wang 3315 33-ASR Teletype, Model TBE
  - Keyboard Input
  - Typed Output at 10 cps
  - Paper Tape Reader - 10 cps
  - Punch Tape Punch - 10 cps
  - Off-line paper tape preparation, reproduction, listing
  - Remote phone line terminal capability

  Wang 1103A Acoustic Coupler (with Automatic Answer-Back)
  - To interface the 3300 for remote phone line connection

  Wang 1104A Acoustic Coupler
  - To interface the 3315 teletypes for remote phone line connection

  Wang 3320 Magnetic Tape Cassette Drive Pair
  - Tape Read 800 cps
  - Tape Write 800 cps
  - Rewind Speed 1 minute
  - Tape Capacity 300,000 byte (max)

III. CPU REGISTER ORGANIZATION

Figure 1, a block diagram of the 3300 CPU organization, illustrates the data storage registers, memory and input/output control logic. The basic storage unit of memory is a byte (8 bits). All registers are 8 bits in length.

The A, Z, S, B, C, and I registers can be addressed or reset by computer instructions. The A and Z registers are general accumulator registers. The S register contains operational status and control bits. The B and C registers are the program instruction counter. The I register contains I/O status information.

The remaining control logic and registers are used in the execution of computer instructions, memory access and input/output. They are not addressable.

All arithmetic operations are performed in the Arithmetic Logic Unit. Arithmetic operations are performed in both binary and decimal mode. Binary arithmetic is performed in 2's complement where one byte assumes a value of from 0 to 255. In decimal mode, a byte containing two 4-bit decimal digits is operated on in decimal arithmetic. In decimal mode, a byte can assume values from 00 to 99.

Information is fetched from memory in a one-byte path via the M register and the L-Bus and M-Bus to various registers. Information is transferred to memory in a one byte path via the Z bus. The memory address is contained in the M and N registers from where it is fetched and decoded.

Information is transferred between input/output device control logic and either the A register or directly to and from memory via the I/O Bus, L-Bus, Z-Bus, and Arithmetic Logic Unit.

The following section contains a detailed description of the CPU registers.

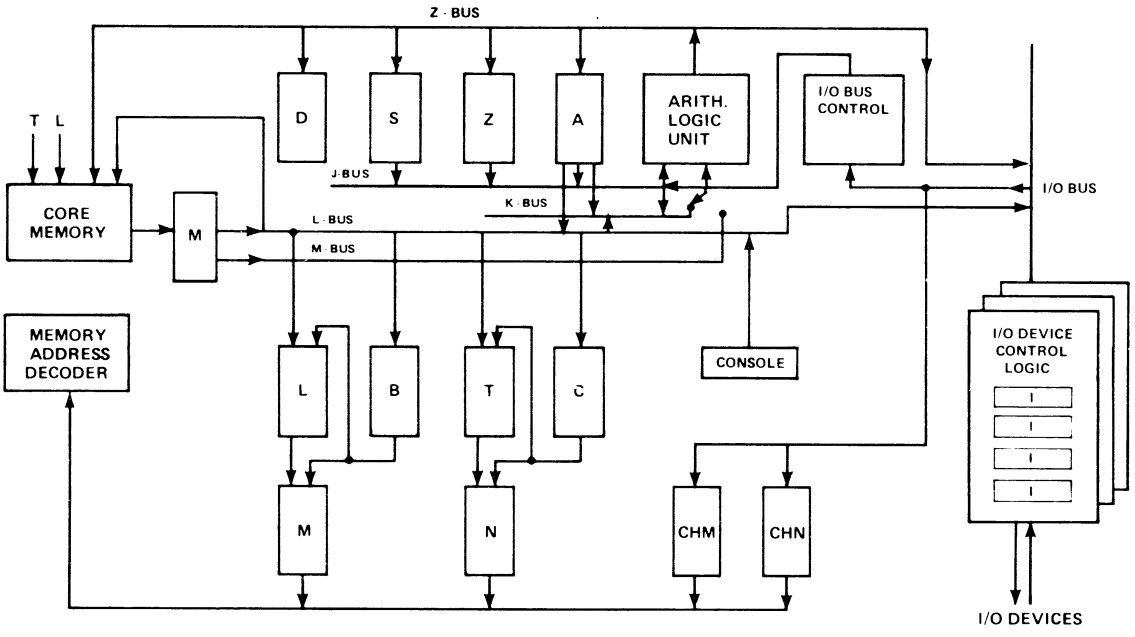
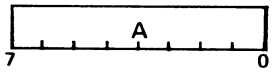


FIG. 1. BLOCK DIAGRAM OF THE WANG 3300 CPU ORGANIZATION

A. Addressable Registers

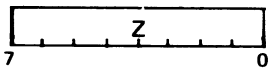
The following registers can be addressed or modified by computer instructions:

(1) **A-Register**



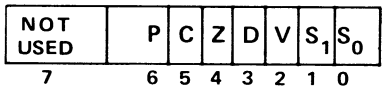
The A-register is an 8 bit general purpose register used primarily as the arithmetic accumulator, and for logical operations, bit manipulation and testing.

(2) **Z-Register**



The Z register is an 8 bit general purpose register. It is independently addressable, but is not generally used in arithmetic, logical, and testing operations like the A-register. It is however treated as a high order extention of the A-register for double byte commands (Double load, Double unload, Double add to Memory, Double shifts, etc.).

(3) **S-Register**
(Status Register)



The Status Register is made up of 7 bits which act and are set independently of one another, but may be addressed as a group with certain status register instructions. Two status bits, P and D, control absolute addressing and arithmetic modes. Others, such as C, Z, and V, are set after the execution of certain instructions to indicate the final results of the instruction operation.

The status register bits are:

P (6) - Absolute Page Indicator (S_6)

P = 0 Page 0
P = 1 Page 1

The setting of the P bit controls what memory page is referenced by memory reference commands which are set up in an absolute page addressing format. If P is set to 0, page zero is referenced (locations 0 to 255). If P is set to 1, Page 1 is referenced, (locations 256 to 511). P is set and reset by status register instructions only, (ONS, OFS, TSAJ, TASJ).

C (5) - Carry Bit (S_5)

C = 1 Carry
C = 0 No Carry

The C status bit (carry bit), is set after the execution of a number of arithmetic and comparison commands. The value of the carry from the high order bit or digit of the last arithmetic operation sets C, (TO 0 or 1). The previous setting of C is added into the lower order bit or digit for arithmetic instructions with carry.

Z (4) - Last Value Zero (S_4)

Z = 1 Result Zero
Z = 0 Result Non-zero

The Z status bit is set after the execution of most load, register transfer, arithmetic, compare and logical instructions. Z is set to 1 if the final result of the arithmetic, logical or load operation was a zero, (result of both bytes for double commands). If it is set to 0, the result was not zero.

D (3) - Decimal Arithmetic Mode Indicator (S_3)

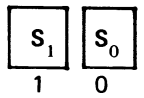
D = 0 Binary Mode
D = 1 Decimal Mode

The setting of the D status bit controls the arithmetic mode for arithmetic instructions. If D is set to 0 binary arithmetic is performed, if it is set to 1 decimal arithmetic is performed. Addressing arithmetic and compare instruction arithmetic will always be done in binary mode, regardless of D. D is set and reset by status register instructions only (ONS, OFS, TSAJ, TASJ).

- Valid Decimal (S_2)

V = 0 Result valid decimal no.
V = 1 Result not valid decimal no.

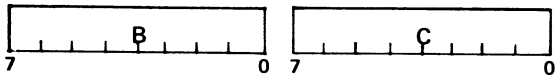
The V status bit is set after the execution of many load, register transfers, arithmetic and compare instructions. The V status bit is set only when the computer is in decimal arithmetic mode (the D status bit = 1). V is set to 0 if the final result of the instruction was a valid decimal number (each 4 bit groups are 0 to 9), and it is set to 1 if it was not.



- User Status Bits (S_1, S_0)

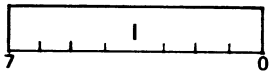
The S_0 and S_1 bits of the status register are general purpose bits which can be set at the programmer's discretion for program control functions. They are set by status register instructions and typically tested with status register skip instructions.

(4) **Program Counter Registers (B, C)**



These two registers contain the 16-bit address of the next instruction to be executed. The B register (most significant 8 bits) contains the page address, and the C register (least significant 8 bits) contains the address within page (0 to 255). A full address of from 0 to $65,535_{10}$ can be expressed. The program counter is set by jump and skip commands.

(5) **I/O Register**

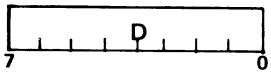


The I/O control logic for each peripheral device contains the equivalent of this 8 bit register. The I/O Register bits indicate various conditions for that device, (ready, error, power off, etc.). The I/O register bits are always sensed with the address of the device stored in the Z-register. These bits are normally reset by I/O clear or enable operations.

B. Non-Addressable Registers

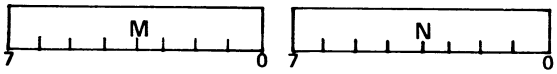
The following registers control and contain information for the execution of program instructions and memory access operations, but are not addressable by computer instructions.

(1) **Instruction Register (D)**



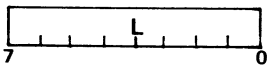
The 8 bit instruction register holds the instruction code byte of the instruction about to be executed.

(2) **Operand Address Registers (M, N)**



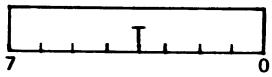
The M and N registers hold the full 16 bit address of the next byte to be fetched from or stored into memory. When instructions are fetched, M and N are set to the current value of the program counter. When a memory reference instruction is executed, M and N receive the final effective operand address.

(3) **L-Register**



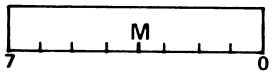
This 8-bit register acts as an intermediate data transfer unit for memory reference, and register transfer operations.

(4) **T-Register**



This 8-bit temporary register is used as an intermediate data transfer unit for memory reference and register transfer operations.

(5) **M-Register**



The M-Register functions as a memory data buffer, for Memory fetches. All data fetched from memory is temporarily stored in the M-register.

(6) **Arithmetic Logic Unit**

This unit performs all arithmetic and logical operations and acts of a data transfer unit for register, memory reference and I/O operations.

(7) **I/O Bus Control Unit**

This unit controls the input/output transfer of data passing on the I/O Bus between the CPU and Device Control logic. It contains the G-register (General I/O) information.

(8) **CHM, CHN**

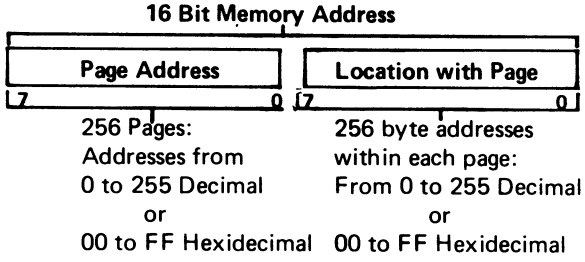
These registers contain the memory address for channel I/O operations.

(9) **Input/Out Device Control Logic**

The units control the operation for one or more I/O devices. The typical control unit also contains device data buffers and the I/O registers (I registers) for each controlled device.

IV. MEMORY ORGANIZATION AND ADDRESSING

The basic unit of addressable memory is a byte, (8 bits). Memory is logically divided into 256 bytes groups called pages. Figure 2 illustrates the format of 3300 memory in pages. A full 16 bit address can reference byte addresses from 0 to $65,535_{10}$. The 16 bit address can also be considered to consist of a high order 8 bits which express the page address and a low order 8 bits which express the byte location within that page:



A. Current Page Addressing

A Wang 3300 instruction is made up of two bytes, (16 bits). For memory reference instructions, the first byte contains the instruction operation code and information defining the mode of addressing. The second byte generally contains an operand address or jump address. Therefore, the second 8-bit byte of the instruction can normally refer to the 256 locations within the current page in which the instruction is located. This is called current page addressing.

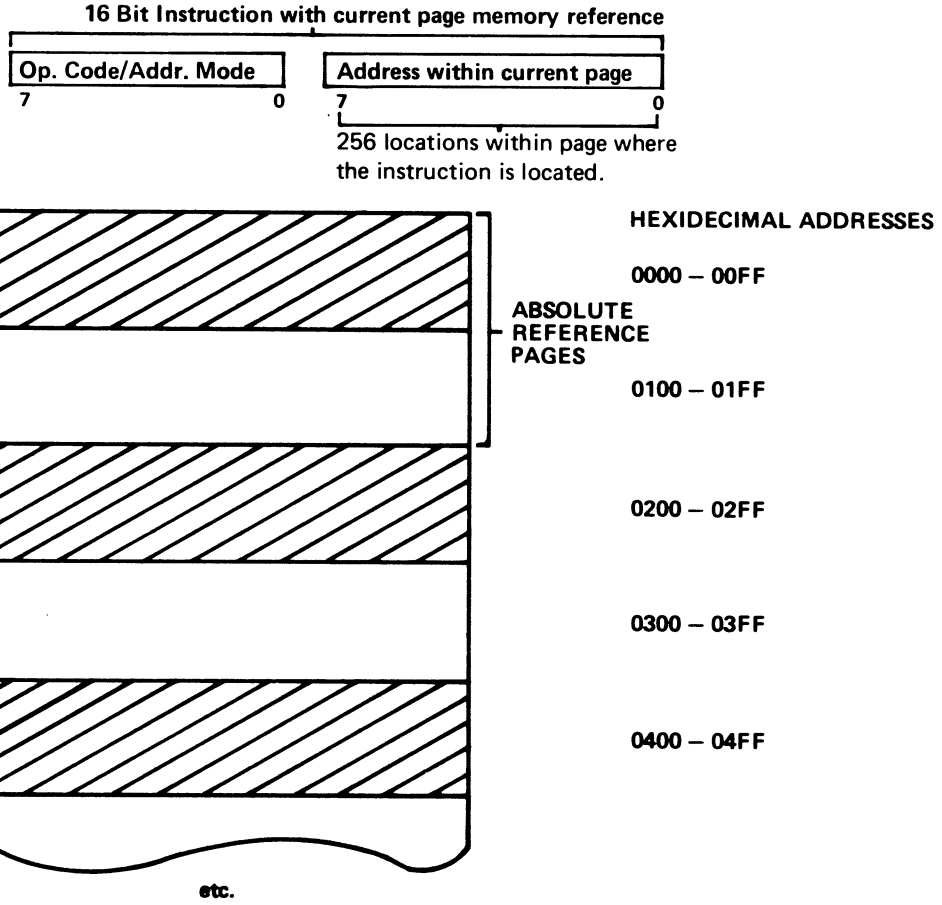
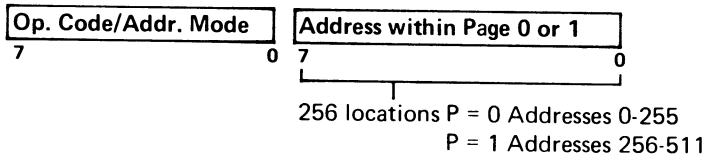


FIG. 2. WANG 3300 MEMORY PAGE FORMAT

B. Absolute Page Addressing

Absolute page addressing is similar in format to current page addressing. However, the 8-bit address portion of the instruction indicates a location within page 0 or page 1 which is being referenced instead of the current page where the instruction is located. Page 0 is referenced if the P status register bit is currently set to 0, page 1 is referenced if P is set to 1. Therefore, an instruction located anywhere in memory can directly reference these two pages.

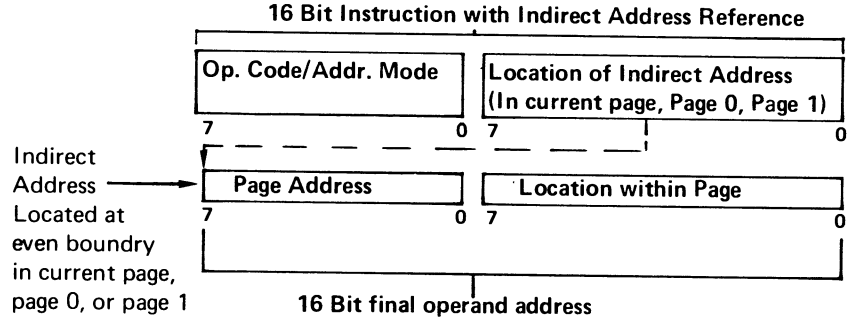
16 Bit instruction with Absolute Page Reference



C. Indirect Addressing

In the indirect addressing mode, the 8 bit address portion of the instruction specifies the location (within the current page, or page 0 or page 1) where the full 16 bit address of the operand is located. Thus, any location in memory can be referenced indirectly. The indirect address must be located on an even address boundary (at an even address). It acts as a pointer to a second location where the final operand is located.

16 Bit Instruction with Indirect Address Reference



D. Auto-Index Indirect Address Reference

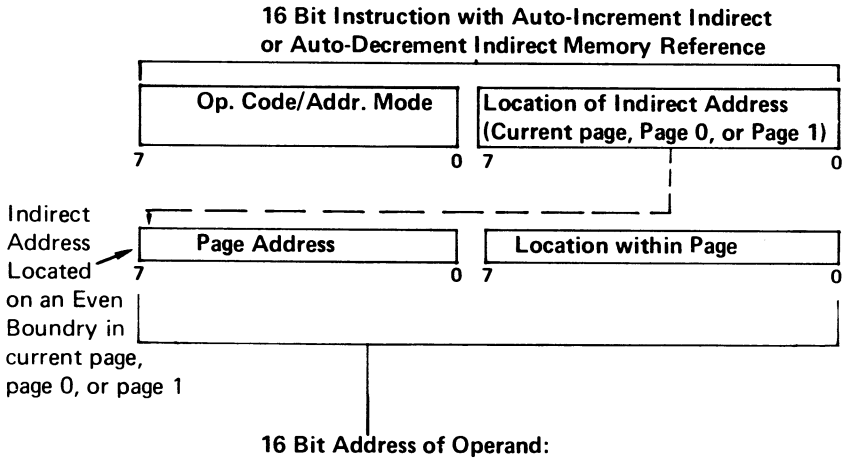
A group of Wang 3300 memory reference instructions have in addition to normal indirect addressing capability, a mode whereby the indirect address is incremented or decremented after or before it is referenced indirectly. This permits lists or tables to be processed efficiently without the bookkeeping generally necessary with indexing operations. The following two modes of auto-index indirect addressing are available:

(1) Auto Incrementing Indirect Mode

An operand is referenced via an indirect address. After the indirect address is referenced to obtain the operand, it is incremented. For single byte memory reference command the final value of the indirect address is one greater than the original value. For double byte memory reference commands, the final value is two greater. The indirect address must be located on an even boundry. In addition, for double byte memory reference commands the original value of the indirect address must also be even.

(2) **Auto Decrementing Indirect Mode**

An operand is referenced via an indirect address. Before the indirect address is referenced to obtain the operand, it is decremented. For single byte memory reference commands, the final value of the indirect address is one less than the original value. For double byte memory reference commands, the final value of the indirect address is two less than the original value. The indirect address must be located on an even boundry. In addition, for double byte memory reference commands, the original value of the indirect address must also be even.

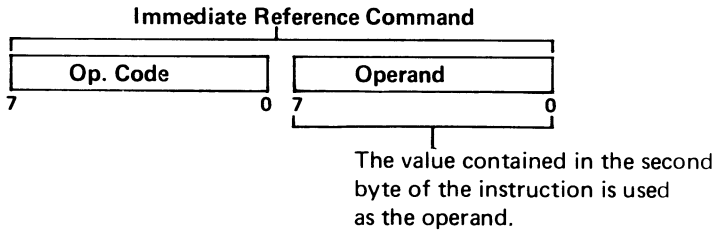


(a) Auto Increment, single byte operand: +1 After reference
(b) Auto Increment, double byte operand*: +2 After reference
(c) Auto Decrement, single byte operand: -1 Before reference
(d) Auto Decrement, double byte operand*: -2 Before reference

*Original value of indirect address must be even.

E. Immediate Reference

The Wang 3300 has a group of immediate reference commands. In this mode the second byte of the instruction is treated as the operand and is loaded or used in arithmetic, logical, and comparison commands.

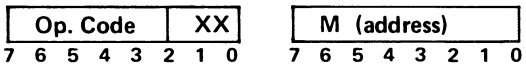


V. INSTRUCTION FORMATS

All Wang 3300 instructions occupy two bytes of memory. They can be classified into the following six formats:

A. Format 1 - Memory Reference Instructions (Without Auto-Index)

This format class includes all memory reference instructions which cannot be used with auto-indexing. For this format the left six bits of the operation code byte contain the op code, the right two bits specify the addressing mode. The second byte of the instruction contains the 8 bit address of the operand or indirect address within the current or an absolute page, (page 0 or page 1). For absolute page reference, page 0 is selected if the P status register bit is 0, page 1 is selected if the P status register bit is set to 1.



FORMAT 1

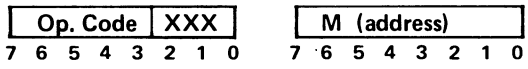
Memory Reference Mode

XX = 00 absolute page direct
XX = 01 absolute page indirect
XX = 10 current page direct
XX = 11 current page indirect

M = Address of the operand or indirect address operand within the current page or an absolute page

B. Format 2 - Memory Reference Instructions (With Auto-Index)

This format class includes all memory reference instructions that can be used in auto-index addressing mode. In this format, the left 5 bits of the OP Code byte contain the Op Code, the right 3 bits specify the addressing mode. The second byte of the instruction specifies the 8 bit address of either the operand or indirect address within the current or an absolute page.



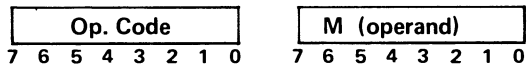
Memory Reference Mode

XXX = 000 absolute page, direct
XXX = 001 absolute page, indirect
XXX = 010 current page, direct
XXX = 011 current page, indirect
XXX = 100 absolute page, indirect with auto-increment
XXX = 101 absolute page, indirect with auto-decrement
XXX = 110 current page, indirect with auto-increment
XXX = 111 current page, indirect with auto-decrement

M = Address of Operand or Indirect Address with the Current or an Absolute Page

C. Format 3 - Immediate Reference Instructions

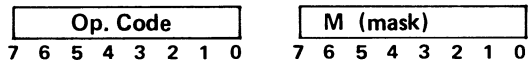
In this format class, the second byte of the instruction is used as the operand. It is either loaded into the Z or A registers or used directly in arithmetic, logical, or comparison operations. The entire 8 bits of the operation byte is used for the OP Code.



M is used as the operand

D. Format 4 - Register Setting and Skip Instructions

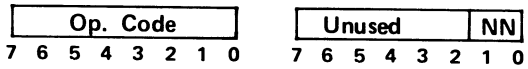
In this class, the second byte of the instruction is generally used as a mask to specify which bits of a particular register will be tested by a skip instruction or set.



M = Mask to specify what bits in a register are to be set or tested.

E. Format 5 - Shift/Rotate Instructions (A-Register)

This format class includes A-register shift and rotate instructions. The rightmost two bits of the second instruction byte specify the shift count. The remaining bits in this byte are unused.

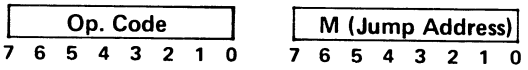


NN = Shift Count

NN = 00 Shift 1 bit
NN = 01 Shift 2 bits
NN = 02 Shift 3 bits
NN = 03 Shift 4 bits

F. Format 6 - Mini Instructions (Current Page Jump)

In this format class, the instruction generally performs register manipulations or input/output, and does not explicitly require the second instruction byte. It is used in all cases to perform a jump to any other location within the current page of the instruction.



M specifies the current page address of the next instruction to be executed.

VI. WANG 3300 INSTRUCTION REPERTOIRE

The Wang 3300 has 68 instructions. Each instruction occupies two bytes of memory. The instruction can be classified into the following functional groups:

- Memory Reference Group (Without Auto-Index)
- Memory Reference Group (With Auto-Index)
- Immediate Reference Group
- Shift and Rotate Group
- Conditional Jump Group
- Conditional Skip Group
- Register Transfer And Manipulation Group
- Input/Output And Interrupt Group

The instructions are described in detail in this section by functional group. Along with the op code and instruction description, the instruction timing and status bits effected are listed. The timing is given in memory cycles. Each memory cycle takes 1.6 μ sec. In instances where the instruction can use indirect address reference, both the direct and indirect timing are given. All indirect address memory references require two additional memory cycles.

For example, if an instruction lists the following timing:

3, 5 memory cycles

It will take: 3 memory cycles or 3 (1.6) = 4.8 μ sec. to execute with direct memory reference
and
5 memory cycles or 5 (1.6) = 8.0 μ sec. to execute with indirect memory reference

The status bits effected by the execution of the instruction are listed for each instruction. For example:

Status – C, Z, V

Means that the instruction when executed will set the C (carry), Z (Last Result Zero), and V (Last Result Valid Decimal) Status Register Bits. The status register bits which can be set by instruction execution (other than status register instructions) are:

C – Carry
Z – Last Result Zero
V – Last Result Valid Decimal (Set only in Decimal Arithmetic Mode)

A. Memory Reference Group (Without Auto-Index)

		Bits 0 and 1 of Instruction Code Byte	Assembler Symbolic Representation
Addressing Modes:	• Absolute Page, Direct	XX = 00	BA TAGO
	• Absolute Page, Indirect	XX = 01	BA* TAGO
	• Current Page, Direct	XX = 10	BA TAG
	• Current Page, Indirect	XX = 11	BA* TAG

Notes:
- The indirect address must be located on an even address boundary.
- Memory reference instructions which reference a current page address and require more than two memory cycles for execution will not function properly when located in the last location of a page, (XXFE).

BA *Boolean And*

010010XX	M	3, 5 Memory Cycles	Status – Z, V
7 0	7 0		

The contents of the effective memory address are anded with the contents of the A-Register; the result is left in A.

BO *Boolean Or*

010110XX	M	3, 5 Memory Cycles	Status – Z, V
7 0	7 0		

The contents of the effective memory address are ored with the contents of the A-Register, the result is left in A.

BX *Boolean exclusive or*

010100XX	M	3, 5 Memory Cycles	Status – Z, V
7 0	7 0		

The contents of the effective memory address are exclusive ored with the A-Register, the result is left in A.

INC *Increment Memory*

010000XX	M	3, 5 Memory Cycles	Status – Z, V
7 0	7 0		

The contents of the effective memory address is incremented by one, addition is always binary.

JEI *Jump and Enable Interrupt*

010101XX	M	2, 4 Memory Cycles	Status – None
7 0	7 0		

The CP interrupt bit is set to zero, (Enable CP Interrupt) and transfer is made to the effective memory address. (CP Interrupt is disabled when an interrupt occurs or by a DSIJ instruction). The CP interrupt bit, when set to 1, inhibits all interrupt.

JMP *Jump*

010001XX	M	2, 4 Memory Cycles	Status – None
7 0	7 0		

Program Control is transferred to the effective memory address.

JST *Jump and Store Location*

010011XX	M	4, 6 Memory Cycles	Status – None
7 0	7 0		

The 16 bit address of the next sequential instruction is stored in the effective memory address and effective memory address plus one. Program control is transferred to the effective memory address plus two.

B. Memory Reference Group (With Auto-Index)

Addressing Mode:	Bits 0, 1, 2 of Instruction Code Byte	Assembler Symbolic Representation
• Absolute Page, Direct	XXX = 000	(ADD TAGO)
• Absolute Page, Indirect	XXX = 001	(ADD* TAGO)
• Current Page, Direct	XXX = 010	(ADD TAG)
• Current Page, Indirect	XXX = 011	(ADD* TAG)
• Absolute Page, Indirect, Auto-Increment	XXX = 100	(ADD+ TAGO)
• Absolute Page, Indirect Auto-Decrement	XXX = 101	(ADD– TAGO)
• Current Page, Indirect Auto-Increment	XXX = 110	(ADD+ TAG)
• Current Page, Indirect, Auto-Decrement	XXX = 111	(ADD– TAG)

Notes -

- Indirect Addresses must be located on even address boundaries
- For double operand commands, auto-indexing will result in the indirect address being incremented or decremented by 2.
- For Auto-Index Indirect Addressing, Incrementing will occur after the fetch, decrementing will occur before the fetch.
- For Double operand commands, the effective memory address (final address of the operand) must be located on an even address boundary.
- Memory reference instructions which reference a current page address and require more than two memory cycles for execution will not function properly when located in the last location of a page, (XXFE).

ADD *Add*

10000XXX	M	3, 5 Memory Cycles	Status – V, Z
7 0	7 0		

The contents of the effective memory address are added to the A-Register. The carry status bit is not affected. The addition will be made in binary mode if the D status register bit is 0, or in decimal mode if D is 1.

AC *Add with Carry*

10001XXX	M	3, 5 Memory Cycles	Status – V,Z,C
7 0	7 0		

The contents of the effect memory address and the C status register bit (carry) are added to the A-Register. The C status register bit will be set to the new value of the carry resulting from the addition, (0 or 1). The addition will be made in binary mode if the D status register bit is 0, or in decimal mode if D is 1.

AMC *Add to Memory with Carry*

10011XXX	M	3, 5 Memory Cycles	Status – V,Z,C
7 0	7 0		

The contents of the A-Register and the C Status Register bit (carry) are added to contents of the effective memory address. The C Status Register bit will be set to the new value of the carry resulting from the addition, (0 or 1). The addition will be made in binary mode if the D status register bit is 0, or in decimal mode if D is 1.

C *Compare*

10010XXX	M	3, 5 Memory Cycles	Status – Z,V,C
7 0	7 0		

The 2's complement of the contents of the A-Register and the contents of effective memory address are added together in binary mode. The result is not saved but the Z (result zero) and C (carry) status register bits are set. The comparison jumps JGT, JNE, JLT, JEQ can be used subsequently.

DAM *Double Add to Memory*

10101XXX	M	4, 6 Memory Cycles	Status – Z,V,C
7 0	7 0		

The contents of the Z and A registers and the C status register bit (carry) are added to the two byte contents of the effective memory address and memory address + 1. The C status register bit (carry) is set to the new carry resulting from the two byte addition. The addition will be made in binary mode if the D status register bit is 0, or in decimal mode if D is 1. The Z status bit reflects both bytes of the result. The V status bit reflects only the high order byte.

DCM *Double Compare with Memory*

10100XXX	M	4, 6 Memory Cycles	Status – C,Z,V
7 0	7 0		

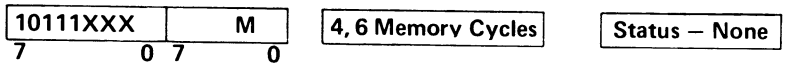
The 2's complement of the contents of the Z and A registers and the two byte contents of the effective memory address, and memory address + 1 are added together. The result is not saved but the Z (result zero) and C (carry) status register bits are set. The comparison jumps, JGT, JNE, JLT, and JEQ can be used subsequently. The Z status bit reflects the result of both bytes. The V status bit reflects only the high order byte.

DL *Double Load*

10110XXX	M	4, 6 Memory Cycles	Status – Z, V
7 0	7 0		

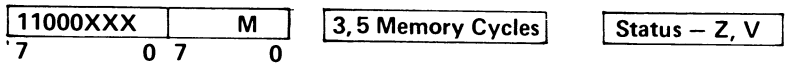
The contents of the effective memory location is loaded into the Z register, the contents of the effective memory location + 1 is loaded into the A-register. The V status bit reflects only the high order byte loaded, (final contents of Z register), while the Z status bit reflects the result of both bytes.

DU *Double Unload*



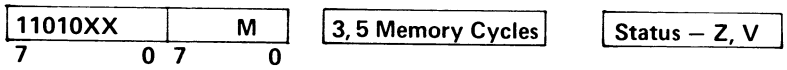
The contents of the Z register replaces the contents of the effective memory address, the contents of the A-register replaces the contents of the effective memory address + 1.

LA *Load A*



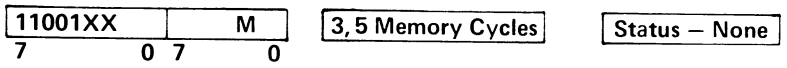
The contents of the effective memory address is loaded into the A-Register.

LZ *Load Z*



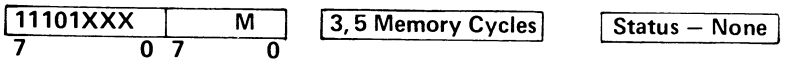
The contents of the effective memory address is loaded into the Z Register.

UA *Unload A (Store A)*



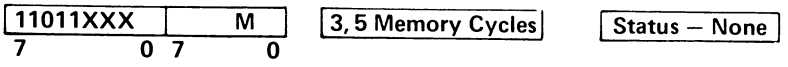
The contents of the A-Register replace the contents of the effective memory address.

UAH *Unload A High Digit*



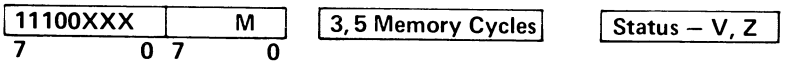
Bits 7 - 4 of the A register replaces bits 7 - 4 of the contents of the effective memory address.

UZ *Unload Z (Store Z)*



The contents of the Z register replace the contents of the effective memory address.

XMA *Exchange Memory and A*

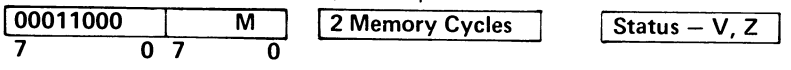


The contents of the effective memory address is exchanged with the contents of the A-register. (V, Z status bits reflect final contents of the A-register.)

C. Immediate Reference Group

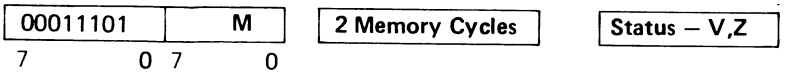
All instructions use the second byte of the instruction, M, as an operand.

AI *Add Immediate*



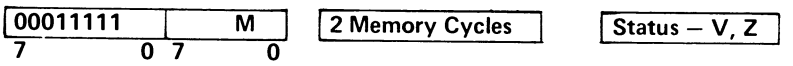
The second byte of the instruction, M, is added to the A register. The status register C bit (carry) is not affected. The addition is performed in binary mode if the D status bit is set to 0, or in decimal mode if D is 1.

BAI *Boolean AND Immediate*



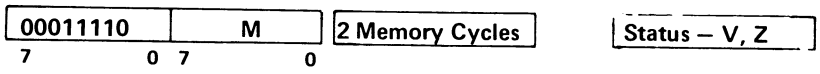
The second byte of the instruction, M, is anded with the contents of the A-register, the result is stored in A.

BOI *Boolean OR Immediate*



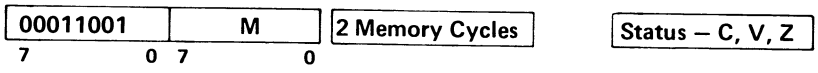
The second byte of the instruction, M, is ored with the contents of the A-register, the result is stored in A.

BXI *Boolean Exclusive or Immediate*



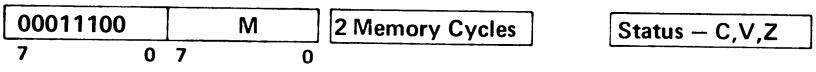
The second byte of the instruction, M, is Exclusive Ored with the contents of the A-register, the result is stored in A.

CI *Compare Immediate*



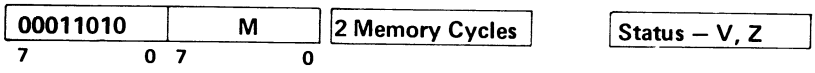
The 2's complement of the contents of A and the contents of the second byte of the instruction, M, are added together. No result is saved but the Z (result zero) and C (carry) status register bits are set. The comparison jumps JGT, JNE, JLT and JEQ can be used subsequently.

DLI *Double Load Immediate*



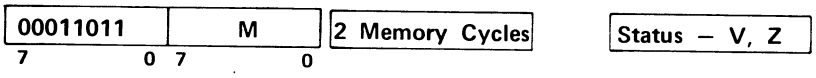
The second byte of the instruction, M, replaces the contents of the A-register. The high order bit of M, (M_7), is propagated thru the Z-register. The C status bit, (carry) is set to 0.

LAI *Load A-Register Immediate*



The second byte of the instruction, M, replaces the contents of the A-register.

LZI *Load Z Register Immediate*



The second byte of the instruction, M, replaces the contents of the Z-register.

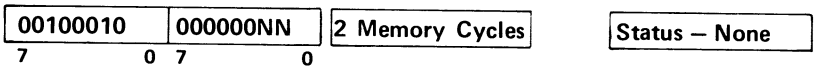
D. Shift and Rotate Group

For all instructions in this group except SDJ, SBJ, and SBCJ the low order two bits of the address instruction byte (M_0 and M_1), specify the shift count which can be 1 to 4 bits. The shift count for SDJ, SBJ, SBCJ is explicitly 1 bit or 1 digit (4 bits).

For all single register shift and rotate instructions the shift count, NN, is interpreted as follows:

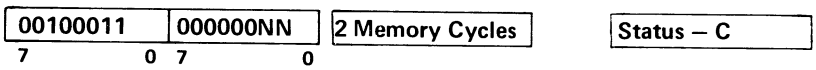
NN = 00 Shift 1 bit
NN = 01 Shift 2 bits
NN = 10 Shift 3 bits
NN = 11 Shift 4 bits

RT *Rotate A Left*



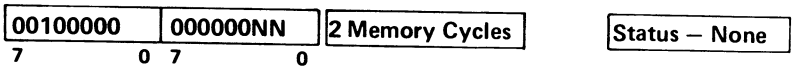
The contents of the A-register is circularly rotated left NN bits, (1-4). Bits shifted out of A_7 replace A_0.

RTC *Rotate A Left With Carry*



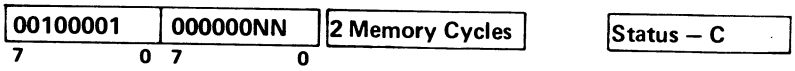
The C status register bit, (carry) is treated as an extension of the A-register. The contents of these 9 bits are circularly rotated left NN bits, (1-4). Bits shifted out of A_7 replace C, bits shifted out of C replace A_0.

SH *Shift A Left*



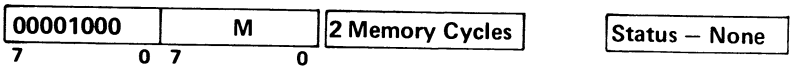
The contents of the A register are shifted left NN bits, (1-4). Zeroes are shifted into the low order bit of A, (A_0).

SHC *Shift A Left With Carry*



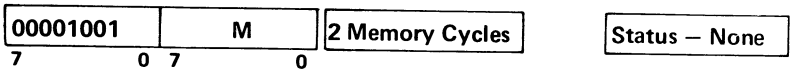
The C status register is treated as an extension of the A register. The contents of these 9 bits are shifted left NN bits, (1-4). Zeroes are shifted into the C bit, the C bit is shifted into A_0.

SBJ *Shift Binary Double And Jump*



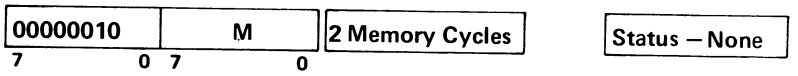
The contents of the Z and A registers are treated as one 16-bit register, and are shifted left 1 bit. Zeroes are shifted into A_0. Program Control is then transferred to the current page address specified by M.

SBCJ *Shift Binary Double With Carry And Jump*



The contents of the Z and A register and the C status register bit (carry) are treated as a 17-bit register and are shifted left 1 bit. ($A_7 \rightarrow Z_0$, $C \rightarrow A_0$). Program Control is then transferred to the current page address specified by M.

SDJ *Shift Decimal Double And Jump*

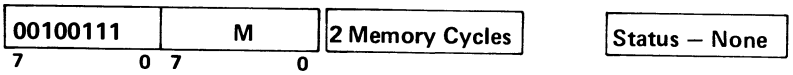


The contents of the Z and A registers are treated as a 16-bit register and shifted left one decimal digit, (4 bits). Zeroes are transferred into the low order 4 bits of A (A_0 - A_3). Program Control is transferred to the current page address specified by M.

E. Conditional Jump Group

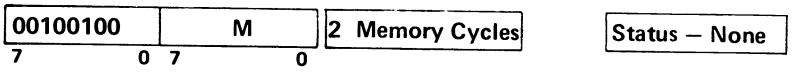
The following instructions produce jumps within the current page depending upon the current setting of the status register Z (result zero) and C (carry) bits. They are typically performed after compare instructions, (C, DCM, CI).

JEQ *Jump If Equal*
(JZ) *(Jump If Result Zero)*



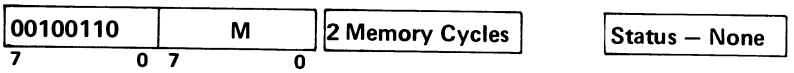
If the Z status register bit (last result zero) is set to 1, (true), Program Control is transferred to the current page address specified by M.

JGT *Jump If Greater Than*
(JNC) *(Jump If No Carry)*



If the C status register bit (carry) is set to 0, (no carry), Program Control is transferred to the current page address specified by M.

JLT *Jump If Less Than*



If the C status register bit (carry) is set to 1, (was a carry), and the Z status register bit (result zero) is set to 0, (last result not zero), Program Control is transferred to the current page address specified by M.

JNE	*Jump If Not Equal*	00100101 (7 0)	M (7 0)	2 Memory Cycles	Status – None
(JNZ)	*(Jump If Not Zero)*				

If the Z status register bit (result zero) is set to 0, (last result not zero), Program Control is transferred to the current page address specified by M.

F. Conditional Skip Group (A-Register, Status Register, I/O Register)

All instructions in this group operate in a similar manner. An 8-bit register or the equivalent is interrogated, (A-register, S-register, or I/O Status Register). The second byte of the instruction acts as a mask to specify which bits in the interrogated register will be examined, (a bit setting of 1 indicates that the bit is to be compared except for false skips, SMA, SFA, SFS, SFS where a setting of zero indicates that a bit is to be compared). If the specified condition is met, the next instruction (two bytes) is skipped.

SAA	*Skip If Any A*	00010010 (7 0)	M (7 0)	2 Memory Cycles	Status – None

The A register bits specified by corresponding 1 bits in M are interrogated. If a 1 is present, the next instruction is skipped.

SMA	*Skip If Mixed A*	00010001 (7 0)	M (7 0)	2 Memory Cycles	Status – None

The A register bits specified by corresponding 0 bits in M are interrogated. If a 0 is present in any compared bit, the next instruction is skipped.

STA	*Skip If True A*	00010000 (7 0)	M (7 0)	2 Memory Cycles	Status – None

The A register bits specified by corresponding 1 bits in M are interrogated. If all 1's are present, the next instruction is skipped.

SFA	*Skip If False A*	00010001 (7 0)	M (7 0)	2 Memory Cycles	Status – None

The A register bits specified by corresponding 0 bits in M are interrogated. If all 0's are present, the next instruction is skipped.

STS	*Skip If True S*	00010100 (7 0)	M (7 0)	2 Memory Cycles	Status – None

The status register bits specified by corresponding 1 bits in M are interrogated. If all 1's are present, the next instruction is skipped.

SFS	*Skip If False S*	00010101 (7 0)	M (7 0)	2 Memory Cycles	Status – None

The status register bits specified by corresponding 0 bits in M are interrogated. If they are all 0, the next instruction is skipped.

STI	*Skip If True I/O*	00110000 (7 0)	M (7 0)	2 Memory Cycles	Status – None

The I/O register for the I/O device whose address is in the Z register is referenced. The register bits specified by corresponding 1 bits in M are interrogated. If all 1's are present, the next instruction is skipped.

SFI *Skip if False I/O*

00110000	M	2 Memory Cycles	Status – None
7 0	7 0		

The I/O register for the I/O device whose address is in the Z register is referenced. The register bits specified by corresponding 0 bits in M are interrogated. If they are all 0, the next instruction is skipped.

G. Register Transfer and Manipulation Group (With Micro Jumps)

The instructions in this group, in most instances, perform explicit operations with the Z, A, and S registers. In most cases the address portion of the instruction, M, is used to specify the address for a current page jump which is an automatic part of the instruction.

AZAJ *Add Z To A And Jump*

00000100	M	2 Memory Cycles	Status – Z,V,C
7 0	7 0		

The contents of the Z and A registers and the C bit are added and stored in A. The carry from the addition is saved in C. Program Control transfers to the current page address specified by M. The addition will be performed in binary mode if the D status register bit is 0, or in decimal mode if D is 1.

DNJ *Double Not And Jump*

00000011	M	2 Memory Cycles	Status – Z
7 0	7 0		

The contents of the Z and A registers are complemented. A 1's complement is performed if the D status register bit is set to 0, or a 9's complement (4 bit groups) if D is set to 1. Program Control transfers to the current page address specified by M. The Z status bit reflects the final result of the A register.

HLTJ *Halt And Jump*

00000000	M	2 Memory Cycles	Status – None
7 0	7 0		

Processing is halted. When processing is restarted, Program Control is transferred to the current page address specified by M.

JZA *Jump Via Z and A*

00001111	00000000	2 Memory Cycles	Status – None
7 0	7 0		

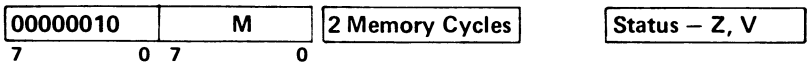
Program Control is transferred to the 16-bit address formed by the current contents of Z and A.

LZAJ *Load Via Z and A And Jump*

00001011	M	4 Memory Cycles	Status – None
7 0	7 0		

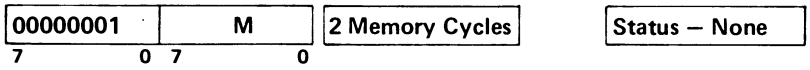
The current contents of the Z and A registers form a 16-bit memory address. The contents of this memory address and the address + 1 are then loaded into Z and A respectively. Program Control is transferred to the current page address specified by M. The memory address specified in Z and A must be even.

NJ *Not A And Jump*



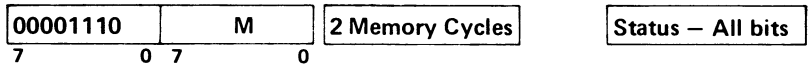
The contents of the A register is complemented. A 1's complement is performed if the D status register bit is set to 0, or a 9's complement (4-bit groups) if D is 1. Program Control is transferred to the current page address specified in M.

PARJ *Parity And Jump*



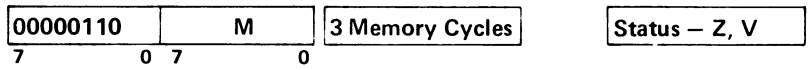
The odd parity bit is generated in bit 7 of the A register based on bits 0-6 of the A register. Program Control is transferred to the current page address specified by M. Bit 7 of the A register must initially be 0.

TASJ *Transfer A To S And Jump*



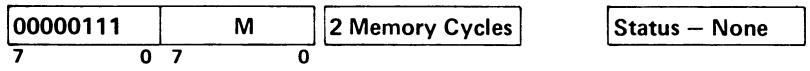
The contents of the A register replace the contents of the S register. Program Control is transferred to the current page address specified by M.

TSAJ *Transfer S To A And Jump*



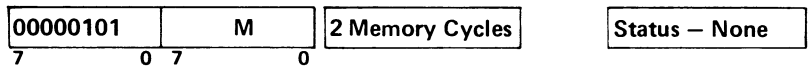
The contents of the S register replace the contents of the A register. Program Control is transferred to the current page address specified by M.

TZAJ *Transfer Z To A And Jump*



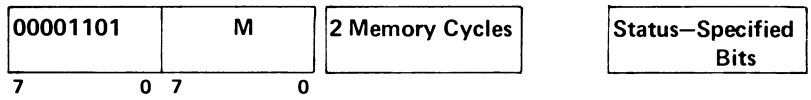
The contents of the Z register replace the contents of the A register. Program Control is transferred to the current page address specified by M.

XZAJ *Exchange Z and A And Jump*



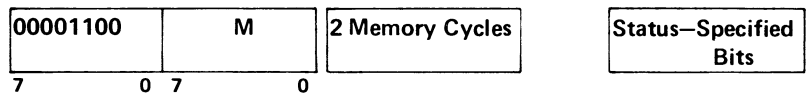
The contents of the Z register are exchanged with the contents of the A register. Program Control transferred to the current page address specified by M.

OFS *Off Status Register*



The status register bits specified by corresponding 0 bits in M are set to 0.

ONS *On Status Register*

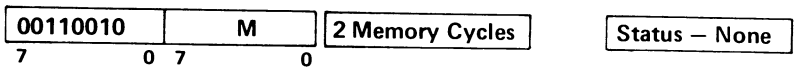


The status register bits specified by corresponding 1 bits in the M field are set to 1.

H. Input/Output And Interrupt Group

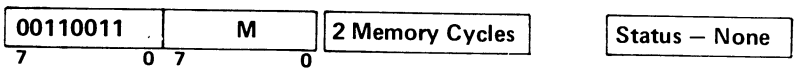
Most of the Input/Output instructions in this group are used in conjunction with a valid device address loaded in the Z-register. In addition, for all instructions in this group, the M portion of the instruction specifies the current page address of an in-page jump which is an integral part of the instruction.

AKIJ *Acknowledge Interrupt And Jump*



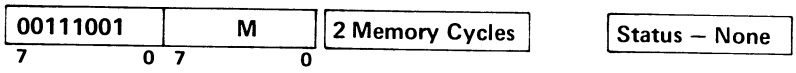
The execution of this instruction, after an interrupt causes the interrupting device address to be transferred to the A-register. Program Control is transferred to the current page address specified by M.

DSIJ *Disable Interrupt And Jump*



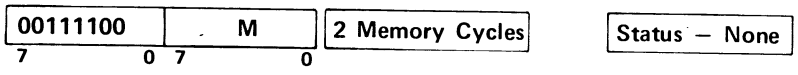
The CP interrupt bit is set to 1, which inhibits all interrupts to the CP. Program Control is transferred to the current page address specified by M. (CP interrupt is automatically disabled when an interrupt occurs).

ONMJ *On Interrupt Mask And Jump*



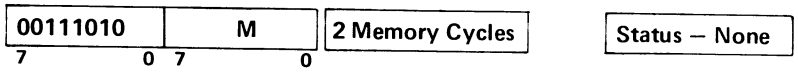
The contents of the A-register are transferred out as a station interrupt mask, one bit per logical I/O station, (bit 0 for station 0, bit 1 for station 1, etc.). If a zero bit is transmitted, interrupt is enabled at the corresponding station, a 1-bit disables interrupt at the station. Program Control is then transferred to the current page address specified by M.

CIOJ *Control Signal To I/O And Jump*



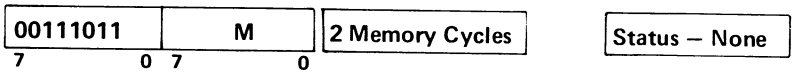
The I/O control command bits in the A-register are transferred to the device whose address is contained in the Z-register. Operations such as clearing and enabling devices, initiating writes or tape motion are performed. Program Control is then transferred to the current page address specified by M.

RDDJ *Read Data And Jump*



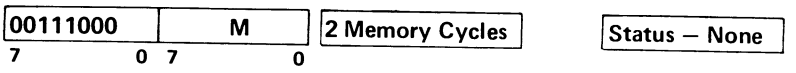
For character buffered I/O devices (I/O bus multiplex mode), one byte (8 bits) is read into the A-register from the character buffer of the device whose address is contained in the Z-register. For certain channel devices, a channel read into memory is initiated for the device specified by the address in Z. In both cases, Program Control is then transferred to the current page address specified by M.

WRDJ *Write Data And Jump*



For character buffered devices (I/O bus multiplex mode), one byte (8 bits) is output from the A-register to the device buffer of the device whose address is contained in the Z-register. For certain channel devices, a channel write from memory is initiated for the device specified by the address in Z. In both cases, Program Control is then transferred to the current page address specified by M.

TIAJ *Transfer I/O to A And Jump*



The I/O status register bits of the device specified by the address in the Z-register is transferred to the A-register. Program Control is then transferred to the current page address specified by M.

VII. INPUT/OUTPUT AND INTERRUPT ORGANIZATION

A. Modes of Input/Output Operation

All input/output operations in the 3300 are carried out through the I/O bus structure. The I/O bus structure has two control modes of operation.

- Multiplex Mode
- Direct Memory Access Channel Mode

In multiplex mode, 8-bit characters are transferred between the CPU A-register and an I/O device buffer in the device control logic. The transfer is made upon execution of a CPU read or write instruction. Ready status information and a corresponding interrupt is signaled when a character has been received into the I/O buffer from a device or output is completed to that device.

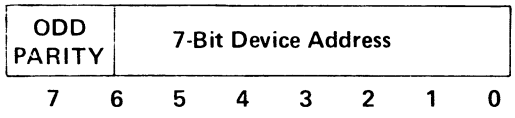
In Direct Memory Access Channel Mode, information is transferred directly between computer memory and device control logic on a cycle stealing basis. When a channel demand is present, the CPU will allocate the next read/write memory cycle to the channel. Accordingly, channel operations are interlaced with CPU instruction executions. In fact, channel read/write memory cycles are allocated to the channel in the middle of an instruction to keep up with high data rates. A read or write command is not required to perform the data transfer, but the device and device control logic must be initialized or enabled prior to the I/O operation. The channel mode of I/O control is used for both low and high speed devices. The Wang 3315 teletype, a low speed terminal device, utilizes I/O Channel logic. Single characters are transferred between fixed memory buffer locations in Page 2 and the device I/O control logic. High speed devices such as discs, utilize the channel logic to asynchronously transfer blocks of characters between memory and the device controller. In this case, the starting memory address, the starting device address and block count are set up by 3300 channel commands (ICH) prior to the transfer, and information can be transferred to and from any computer memory address.

B. I/O Bus Structure Organization

1. Device Addresses

Each I/O device attached to the 3300 has a seven-bit device address. Some devices will have two addresses, one for input and one for output. Up to 128 (2^7) device addresses are available in the 3300. For devices with two addresses, the input device address is even and output device address is odd and one greater than the input address.

All 3300 I/O instructions operate identically in terms of device addressing. The device address is loaded into the low order 7 bits of the Z-register. For certain devices the high order bit of the Z register is set to the odd parity logical sum of the 7 address bits. A device will respond only if the address parity is correct. The I/O instruction which may enable a device, initiate a transfer, or sense I/O device status is then executed. The device address will always be taken from the current contents of the Z-register.



Setup of the Z register prior to an I/O instruction.

2. I/O Bus Stations And Interrupt Priority

For purposes of interrupt, processing priority and control, the I/O bus structure is logically divided into stations. Figure 3 illustrates a typical I/O bus station organization for the 3300. A single device or a number of devices may be attached to a station. Up to 8 stations can be configured on a 3300, controlling up to 128 addressable devices in any combination. I/O operation within the station organization has the following characteristics:

(1) Interrupts and channel cycle stealing demands are sequential. Devices attached on the closest station to the CPU are given highest priority, the second closest, the second highest priority, etc. Lower priority device interrupts are stacked until they can be processed.

(2) For devices on the same station, the ones attached closest to the I/O bus are given the highest priority. Lower priority device interrupts are stacked until they can be processed.

(3) A single station and all devices on that station can be inhibited from interrupt, and subsequently re-enabled by a station interrupt mask instruction (ONMJ).

To summarize,.devices to 3300 I/O bus stations are ordered in 128 levels of sequential interrupt priority on from 1 to 8 stations. All devices assigned to a single station can be inhibited from or enabled for interrupt as a group.

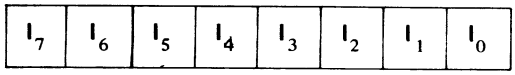
Since the 3300 can be configured with a number of different I/O devices in varying quantities, a variety of station configurations are useful. Typically, high-speed devices will be assigned to higher priority stations or positions on a station, lower speed devices to lower priority stations or positions. Similar devices with identical functions would be assigned to the same station, so that they will receive approximately the same priority and can be inhibited as a group.

C. I/O Status Registers

Figure 3 illustrates the status registers used with Input/Output.

- I/O Status Register - One for each device. They reflect the current operating status of each device.

1. I/O Status Registers



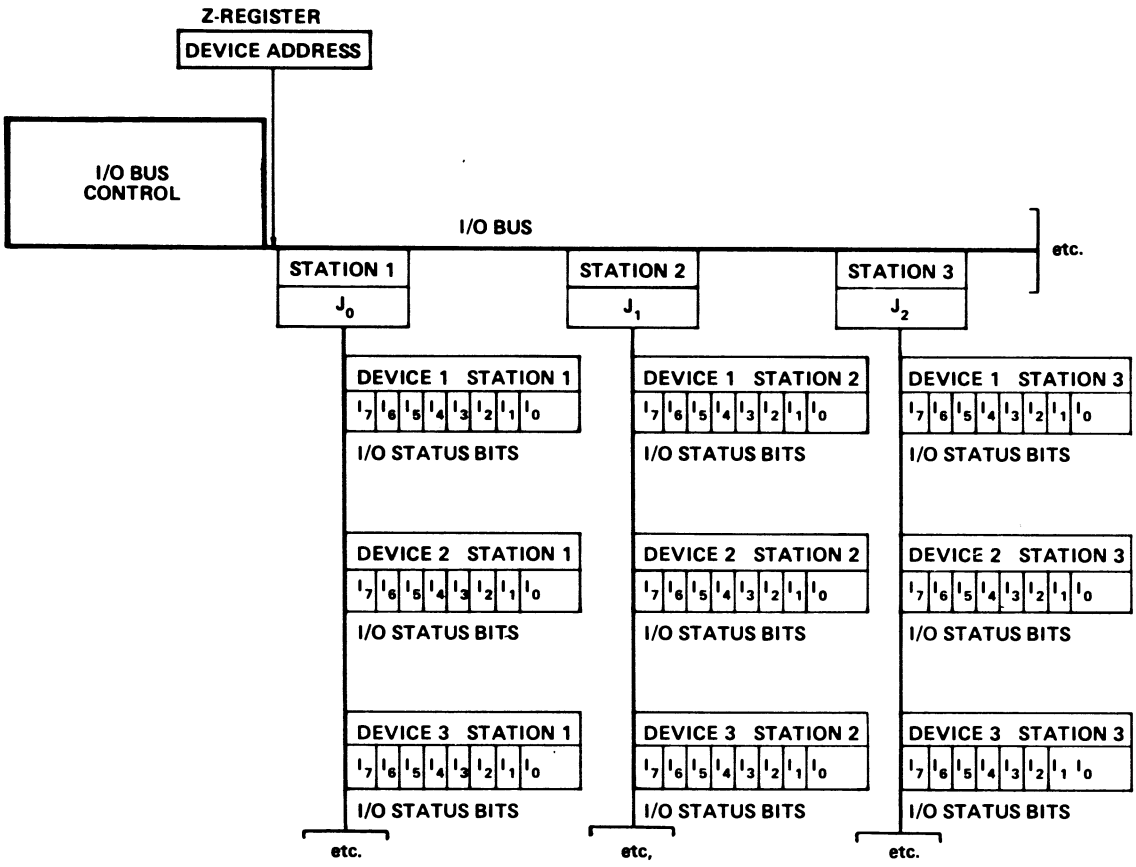
The logic which controls each device contains an 8-bit status register which reflects the current operating status of the device. These device status register bits are sensed by using STI, SFI and TIAJ instructions with the appropriate device address loaded into the Z-register. Typical I/O status bit settings are:

3315 Teletype I_0 - Error

I_7, I_6, I_5, I_4 Ready = 0000 Device Idle and Ready
= 1011 Character Received (Input)
= 1100 Character Transmitted (Output)

The status register bits are sensed to determine ready conditions for non-interrupt I/O and transmission errors. A typical sequence following a write for a 3315 Teletype would be:

```
LZ      DEVADR          Load device address into Z
SFI     B'11000000'     SKIP if ready bits are set
JMP     *-2             Not, continue checking
. . .
. . .
. . .
Continue
```



(1) The Device Address from the Z-register is decoded at each station. If a device is attached to a particular station, I/O command signals and data will be passed thru the station to and from the device.

(2) Each station has an Interrupt Inhibit/Enable Mask bit: J_0, J_1, J_2, etc. They are set or reset by the ONMJ command using the A-register for a mask as follows:

A-register setting for ONMJ

J_7	J_6	J_5	J_4	J_3	J_2	J_1	J_0

where J = 0 Enable
where J = 1 Inhibit

(3) The control logic for each device contains I/O Status Register bits, (I_0 thru I_7). They reflect current device status conditions such as ready, error, etc. They can be sensed by STI, STF, and TIAJ commands with the Z-register set to the device address.

FIG. 3. TYPICAL I/O BUS STATION ORGANIZATION

D. Interrupt Operations and Instructions

As was described in Section B, the 3300 has up to 128 sequential levels of priority interrupt. Priority is established by attaching I/O devices to I/O Bus stations. Stations closest to the CPU and positions on a station closest to the CPU have higher priority. In the station configuration shown in Figure 3, the following priority would be observed.

Station 1	Device 1	Highest Priority
Station 1	Device 2	Next Highest Priority
Station 1	Device 3	Next Highest Priority
Station 2	Device 1	Next Highest Priority
Station 2	Device 2	Next Highest Priority
Etc.		

Interrupt can be inhibited or enabled for all the devices attached to a station. Each station has its own interrupt mask bit. They are illustrated in Figure 3 as J_0, J_1, J_2, etc. The 8 interrupt mask bits for the 8 possible I/O bus stations are set or reset by the ONMJ instruction. For this instruction, the A-register is used as a mask to set and reset corresponding station interrupt inhibit bits, (Bit 0 = station 1, Bit 1 = station 2, etc).

For example:

```
LAI     B'00000101'
ONMJ
```

would inhibit stations 1 and 3, and enable all other stations.

In addition to station interrupt bits, all interrupts to the CPU are inhibited or enabled by a single CPU interrupt bit contained in the I/O Bus Control Unit. The bit is automatically reset to 1, (inhibit), when an interrupt occurs. This prevents other interrupts from occurring during the processing of the current interrupt. When interrupt is completed the interrupt routine can be exited and CPU interrupt re-enabled by executing a JEI instruction, (Jump and Enable CPU Interrupt). The DSIJ instruction (Disable CPU Interrupt and Jump) can be used to inhibit CPU interrupt at any time.

During input/output processing with interrupt, (e.g. station interrupt enabled), an interrupt is requested when an I/O device becomes ready after an input or output operation. If an interrupt is currently being processed, (CPU interrupt temporarily inhibited), the request is stacked until the interrupt priority level associated with the device becomes available.

When the interrupt occurs, program control is transferred to location 2 of page 0. The previous current contents of the program counter, (registers B and C) are then stored in locations 0 and 1 of page 0.

In effect, the interrupt results in a subroutine jump to location 0 of memory. (JST 0). As the interrupt occurs, the CPU interrupt bit is automatically reset inhibiting all other interrupts. As the interrupt routine is entered the current contents of the A, Z, and S registers should be saved by the routine and restored upon exit. The interrupting device is determined by the execution of an AKIJ instruction, (Acknowledge Interrupt and Jump). When this instruction is executed, the device address of the interrupting device is transferred into the A-register, where it may be examined. The last instruction of the interrupt routine should be an indirect JEI command command which re-enables CPU interrupt. A typical I/O interrupt routine is illustrated below.

Typical Interrupt Routine

Location					
Page 0	0,1	TRAP	DC	X'0000'	Address of Interrupted Program Stored here
	1,2		JMP*	IADR	Jump indirectly to Interrupt Routine
	3,4	IADR	DAC	INTR	Address of interrupt routine
	. . .				
	. . .				
	. . .				
Page xx (.	.)	INTR	DU	SVZA	Save the Z-register and A-register
			TSAJ		S-register to A-register
			UA	SAVS+1	Save S-register in the 2nd byte of a ONS instruction
			. . . .		which will restore it.
			. . . .		
			. . . .		
			AKIJ		(Acknowledge Interrupt (Device Address to A-register)
			. . . .		
			. . . .		
			(Establish Interrupting Device Status and set up next I/O)		
			. . . .		
			. . . .		
			DL	SVZA	Restore Z-register and A-register
			OFS	X'FF'	Clear status register to 0
		SAVS	ONS	**	Restore status register
			JEI*	TRAP	Enable CPU interrupt and return to interrupted program. (Indirect jump to saved address).
		SVZA	DC	X'0000'	Location to save Z and A.

E. I/O Instructions

The following I/O instructions are used to perform, test, and control input/output operations on the Wang 3300. In conjunction with each instruction, the Z-register must contain the proper device address of the device being controlled. Both a read or write address are used for certain devices.

CIOJ - Control Signal to I/O and Jump

This instruction transmits the current contents of the A-register to the device control logic. The bit pattern in A produces various control operations to be performed. Typically, functions such as clear device status, enable device for input/output, and initiate write are performed by CIOJ's.

RDDJ - Read and Jump

This instruction initiates read operations for certain devices.

WRDJ - Write and Jump

This instruction initiates write operations for certain devices.

TIAJ - Transfer I/O Register to A-Register and Jump.

This instruction transfers a device I/O status register to the A-register, (for the device selected by the address in Z). The value can then be tested for ready error, etc.

STI - Skip if True I/O Status

This instruction tests specified bits of a selected device I/O status and skips if they are all 1. It is used to test for device ready, error, etc.

SFI - Skip if False I/O Status

This instruction tests specified bits of a selected device I/O status register and skips if they are all 0. It is used to test for device ready, error, etc.

VIII. 3315 TELETYPE TERMINAL OPERATION

A. Operational Characteristics

The standard teletype used with the 3300 is the 33ASR Model TBE. The teletype terminal has both a keyboard/printing unit and a paper tape reader/punch unit. The operation of both units is quite similar. The only differences are:

- For tape input, an 'X-on' character is transmitted to the device to start the tape motion and an 'X-off' character is transmitted to halt tape motion. (For some Model 33ASR's the tape is started by a manual switch.)
- For tape output, the tape On/Off switch on the terminal is manually set to the On position to simultaneously produce a punched tape and printing.

The maximum transfer rate for both units is 10 cps. The major functional characteristics of the 3315 teletype terminal are:

- The teletype has no character buffer in the Control Unit and uses a fixed memory address location in page two for this purpose. Unique addresses are assigned each device. Data transfer is made directly to and from the device and memory without a read or write command. A Wang 3316 teletype control unit controls up to four teletypes in full duplex mode. The control unit transmits data bits serially at 9.09 ms intervals between two predetermined memory locations for each device and the device in a single character channel mode.

- There is no parity error checking capability for the Model TBE teletype I/O. Wang 3300 teletype code is the standard 3300 ASCII code set, (listed in Appendix D of this manual). The parity bit, (the 8th bit) is not used for parity and in general should be stripped and ignored for input. (All eight tracks are used when loading binary system tapes.)
- Teletype I/O operations are normally performed in a full duplex mode. That is, simultaneous input and output operations can occur at the same time and are completely independent. For keyboard input, each received character must be output back to the deivce to produce printing. This echo procedure also verifies correct transmission to the user. (The unit can also operate in half duplex.)

B. I/O Status Register Settings

The I/O status register for each teletype device acts like a counter during the transfer of a character. When an input or output operation is initiated, it is initially set to zero. As the transfer proceeds, bits 4 thru 7 of the register are incremented until it is complete. The I/O status register settings can be sensed by SFI, STI, and TIAJ commands at any time. The status register settings for both read and write operations are listed in Table 1.

TABLE 1 – TELETYPE I/O STATUS REGISTER SETTINGS DURING DATA TRANSFER

TTY INPUT			TTY OUTPUT		
TELETYPE CONTROLLER TRANSFER CYCLE (9.09 MS/CYCLE)	STATUS REGISTER SETTING (HEX)	DESCRIPTION	TELETYPE CONTROLLER TRANSFER CYCLE (9.09 MS/CYCLE)	STATUS REGISTER SETTING (HEX)	DESCRIPTION
	00	Idle – No data transfer occurring		00	Idle – No data transfer occurring
1 thru 9	10 thru A0	Input request received and bits are being transferred serially to the page 2 input memory buffer.	1 thru 10	10 thru B0	Output request made and bits are being transferred serially from the page 2 output memory buffer.
10 thru 11	B0	READY Data transfer complete. An interrupt will be issued if interrupt is enabled.	11	C0	READY Data transfer complete. An interrupt will be issued if interrupt is enabled.
During Interrupt or at the end of cycle 11	00	Status will be automatically reset back to idle at the end of cycle 11 or when an AKIJ command is executed in the interrupt routine.	During Interrupt	00	Status will automatically be set back to idle during an interrupt routine when an AKIJ command is executed.
Any time during cycles 1 - 9	- 1	Read Error (A status clearing CIOJ command was executed during data transfer.)	Any time during cycles 1 - 10	- 1	Write Error (Device power off or a status clearing CIOJ command was issued during data transfer.)

Note: – The input status register for a Teletype is reset to idle (00) by the following:

(1) When an AKIJ command is executed during the interrupt routine (cycles 10 and 11).

(2) At the end of cycle 11. This will also cause a pending interrupt to be cancelled.

(3) When a Clear Status command is executed. (CIOJ with A = 83, Z = Input Address).

Note: – The output status register for a Teletype will be reset back to idle (00) by the following:

(1) When an AKIJ command is executed during the interrupt routine, (cycle 11 or thereafter).

(2) When a Clear Status command is executed. (CIOJ with A = 83 or 08, Z = Output Address.)

A pending output interrupt will never be cancelled unless status is reset by a CIOJ.

C. Read and Write Programming Procedures

1. Keyboard Read Operations

Keyboard read is a self-initializing operation. It is not required that the teletype be enabled for input. When the terminal operator strikes a key, the character transfer is initiated automatically. The following steps are involved in the keyboard read operation:

a. Enable station interrupt with an ONMJ, if interrupt required and not enabled. Enable CPU interrupt with JEI if required and not enabled.

b. Clear the device for input with a CIOJ command.
CIOJ with Z = Device Read Address; A = 83 (HEX)

c. The device input status is set to and remains at 00, (idle) until a key is struck. At this point, the device input status register is incremented until the data transfer is complete, at which time a status value of B0 (HEX) exists. The character will be completely transferred to the appropriate page two memory buffer. If interrupt is enabled, an interrupt request will be set up.

d. When the interrupt occurs, or when ready is sensed in non-interrupt mode, the character may be fetched from the appropriate page two memory buffer and processed. At this time, the character should be written back to the device to print it, (an independent I/O operation). In interrupt mode, status is reset back to 00, (idle) when the AKI instruction is executed in the interrupt routine. The input status is automatically reset back to 00, (idle), 13.5 milliseconds after data transfer is completed (B0 ready status) if not reset by an AKI or CIOJ. If an interrupt has not been executed by this time, the interrupt is lost although the character read will remain in the input buffer. At this time, additional input can be received and steps (c) and (d) are repeated for interrupt mode, steps (b), (c), and (d) are repeated for non-interrupt mode.

2. Teletype Tape Read Operations

Teletype tape read operations are identical with keyboard tape read except that:

- A 'X-on' character is output to the teletype initially to initiate movement of the tape. After all required characters are read, a 'X-off' character is output to the teletype to halt tape motion. (Both are write operations.)
- Read tape characters are generally not echoed back to the Teletype for printing.

The following steps occur for tape read operations:

a. A terminal operator mounts the tape, and sets the reader switch to the "STOP" position.

b. Enable station interrupt with an ONMJ, and CPU interrupt with a JEI, if interrupt is desired.

c. A 'X-on' character is transmitted to the teletype to start tape motion, (a code of 11 HEX).

d. Initially clear the device for input. CIOJ with Z = Device Address, A = 83 (HEX).

e. As each character is transmitted, the device status register is incremented until transfer is complete, status B0 (READY).

f. An interrupt occurs, or ready is sensed and the character is fetched from the memory buffer and processed. Status is automatically set to 00 (idle) 13.5 milliseconds after ready, or by an AKI, and the next character transfer begins. Steps (e) and (f) are repeated.

g. If the last required character has been ready, a 'X-off' code is transmitted to the Teletype to halt tape motion (a code of 13 HEX). Since this takes approximately 100 milliseconds, two additional tape characters will be read before tape input is terminated.

3. Print/Tape Punch Write Operations

Teletype printing and tape punching operations are simultaneous; the only differences being that the tape is manually switched on and off of the terminal. When the punch is on, both printing and punching will occur, while only printing occurs with the punch set to off. Unlike teletype read operations, write operation requires that an output enable and clear (CIOJ) be executed prior to every character transfer. In addition, an interrupt is issued after the transfer cycles are completed at the beginning of cycle 11 (90.91 ms after the start of write). The ready condition and/or interrupt may be processed any time after cycle 11 and is not lost. I/O register status is reset to 00, (idle) either by a CIOJ command with A=83 (Clear), or A=08 (Clear and Write) or by the execution of an AKIJ command in the interrupt routine.

The following steps occur in a teletype write sequence:

a. Enable the device station interrupt with an ONMJ, if interrupt mode is desired and the station is not already enabled. Enable CPU interrupt with a JEI command.

b. Place the desired output character in the appropriate page two memory buffer.

c. Initiate output, CIOJ with Z = Device Write Address, A = 08 (HEX) (Clear and Write). I/O status immediately becomes 10, (write requested).

d. The teletype output status register is then stepped until data transfer is complete, at which time it is set to C0 (interrupt pending, ready). If interrupt is enabled, an interrupt is then requested. When the data transfer is complete, the memory buffer will be cleared to zero.

e. When the interrupt occurs, or when the status is sensed as ready for non-interrupt mode, a new character can be set up and enabled, repeating steps (c) through (e). If an interrupt occurs, status will be set back to 00 (idle) when an AKI command is executed. On output operations, the status is also reset by the execution of a CIOJ command with A = 83 (Clear) or A = 08 (Clear and Write).

D. Program Examples

1. Read a character from keyboard without interrupt

```
READ    LZ      RADR        LOAD READ DEVICE ADDRESS
        LAI     X '83'      CLEAR CODE INTO A CLEAR
        CIOJ                DEVICE IN INPUT MODE
WAIT    STI     X 'B0'      CHECK READY AND WAIT TO
        JMP     WAIT        RECEIVE CHARACTER
        LA*     RBUF        INPUT COMPLETE, GET CHARACTER
                            FROM MEMORY PAGE 2 BUFFER
            (PROCESS CHARACTER)
            .
            .
            .
        JMP     READ        BACK FOR NEXT INPUT
RBUF    DC      X'0220'
RADR    DC      X'20'
```

2. Print a character without interrupt

```
WRITE   LA      CHAR        STORE PRINT CHAR INTO MEMORY
        UA*     WBUF        DEVICE BUFFER (PAGE 2)
        LZ      WADR        LOAD DEVICE ADDRESS
        LAI     X'08'       INITIATE WRITE, CLEAR STATUS
        CIOJ
WAIT    STI     X 'C0'      WAIT UNTIL CHARACTER
        JMP     WAIT        IS OUTPUT, STATUS 0
            (GET NEXT CHAR TO PRINT)
            .
            .
            .
        JMP     WRITE       BACK TO PRINT NEXT CHAR
WBUF    DC      X'0220'     LOCATION OF WRITE BUFFER IN
                            MEMORY (TTY NO. 1)
WADR    DC      X'A1'       WRITE DEVICE ADDRESS (TTY NO.1)
```

3. Input from the Teletype tape unit without interrupt

```
TREAD   LAI     X '11'          STORE X-ON CHARACTER
        UA*     WBUF            IN PAGE 2 DEVICE BUFFER
        JST     TWRIT           OUTPUT CHAR TO TTY
CLEAR   LZ      RADR            LOAD DEVICE READ ADDRESS
        LAI     X '83'          LOAD CLEAR CODE
        CIOJ                    CLEAR TTY IN INPUT MODE
WAIT    STI     X 'B0'          WAIT UNTIL A
        JMP     WAIT            CHARACTER IS READY
        LA*     RBUF            GET CHARACTER
             (PROCESS READ CHARACTER)
        .
        .
        .
        JMP     CLEAR           NOT LAST CHAR, BACK FOR NEXT
        LA      X '13'          WAS LAST CHARACTER
        UA*     WBUF            SETUP X-OFF CHAR IN BUFFER
        JST     TWRIT           AND OUTPUT IT
          (CONTINUE)
        .
        .
TWRIT   DC      X '0000'        ROUTINE TO OUTPUT CHAR TO TTY
        LZ      WADR            WRITE DEVICE ADDRESS IN Z
        LAI     X '08'          CLEAR AND WRITE MASK INTO A
        CIOJ                    INITIATE TTY WRITE
        JMP*    TWRIT           RETURN
RBUF    DC      X '0220'
WBUF    DC      X '0221'
RADR    DC      X '20'
WADR    DC      X 'A1'
```

4. Print or punch on Teletype with interrupt
(SET UP DEVICE IN OUTPUT MODE AND ENABLE INTERRUPT)

```
INIT    LA      MASK            ENABLE INTERRUPT FOR THE STATION
        ONMJ                    TO WHICH DEVICE IS CONNECTED
        JEI      *+2            ENABLE CPU INTERRUPT
        JST     TWRIT           CALL ROUTINE TO SET UP FIRST WRITE
             (CONTINUE WITH PROGRAM
             UNTIL INTERRUPT)
(ROUTINE TO SET UP NEXT OUTPUT CHAR IN PAGE 2 MEMORY BUFFER)
IWRITE  DC      X '0000'
        .       (GET NEXT CHAR TO OUTPUT)
        .
        .
        LA      CHAR            LOAD NEXT OUTPUT CHARACTER
        UA*     WBUF            STORE IT IN DEVICE MEMORY BUFFER
        LZI     X 'A1'          WRITE DEVICE ADDRESS INTO Z
        LAI     X '08'          CLEAR AND WRITE CODE BIT INTO A
        CIOJ                    INITIATE TTY WRITE
        JMP*    IWRITE          RETURN
WBUF    DC      X '0221'
```

(INTERRUPT ROUTINE, WHICH IS ENTERED WHEN THE CHARACTER IS OUTPUT AND DEVICE BECOMES READY. THE INTERRUPT RESULTS IN A FORCED JST TO ADDRESS 0000.

```
Location        ORG     X '0000'    INTERRUPT PROCEDURE AT LOC. 1
 0000   TRAP    DC      X '0000'    ADDRESS OF INTERRUPTED INSTRUC-
 0002           JMP*    IADR        TIONS STORED HERE
 0004   IADR    DAC     INTR + 2    JMP INDIRECTLY TO INTERRUPT
                                    ROUTINE ADDRESS OF INTERRUPT
INTR            DAC     *           ROUTINE
                DU      SVZA        (SAVE Z, A AND S REGISTERS)
                TSAJ                SAVE S REGISTER
                UA      SAVS + 1
                OFS     X 'FF'      SET S REGISTER TO ABSOLUTE PAGE 0
                DL      TRAP        SAVE INTERRUPT RETURN ON CURRENT PAGE
                DU      INTR
                AKI
                        ACKNOWLEDGE INTERRUPT (INTERRUPTING DEVICE AD-
                        DRESS→A) (FROM DEVICE ADDRESS DETERMINE WHAT
                        DEVICE HAS INTERRUPTED, WHETHER THERE IS MORE TO
                        OUTPUT, AND SET UP TO OUTPUT NEXT CHARACTER
                JMP     EXIT        LAST CHARACTER DONE, RETURN
                .
                .
                .
  MORE          JST     IWRITE      MORE OUTPUT NEXT CHARACTER
  EXIT          DL      SVZA        (RESTORE Z,A,S, REGISTERS)
                OFS     X 'ff'
  SAVS          ONS     **
                JEI*    INTR        ENABLE INTERRUPT, RETURN
  SVZA          DC      X '0000'    INTERRUPT PROGRAM
```

5. Read characters from keyboard and echo back, without interrupt.

```
  READ          LZI     X '20'      LOAD DEVICE READ ADDRESS INTO Z
                LAI     X '83'      CLEAR/ENABLE CODE INTO A
                CIOJ                CLEAR READ STATUS
  WAIT          STI     X 'B0'      WAIT FOR INPUT READY STATUS
                JMP     WAIT
                LA*     RBUF        GET CHARACTER READ FROM BUFFER
                LZI     X 'A1'      LOAD DEVICE WRITE ADDRESS INTO Z
                SFI     X 'FF'      CHECK IF STATUS ZERO (IDLE)
                JMP     * + 4       NO
                JMP     REDY        YES, GO TO WRITE
                STI     X 'C0'      WAIT FOR READY STATUS
                JMP     * - 2
  REDY          UA*     WBUF        STORE READ CHARACTER INTO WRITE BUFFER
                LAI     X '08'      CLEAR IWRITE CODE INTO A
                CIOJ                CLEAR STATUS/INITIATE WRITE
                JMP     READ        BACK TO GET NEXT INPUT
  RBUF          DC      X '0220'    LOCATION OF INPUT BUFFER TTY NO. 1
  WBUF          DC      X '0221'    LOCATION OF OUTPUT BUFFER TTY NO. 1
```

E. Teletype Device Buffer Addresses

The following Page 2 memory locations are reserved as read and write buffers for sixteen teletype terminals.

MEMORY BUFFER ADDRESS (HEX)	TERMINAL	DEVICE ADDRESS (HEX) WITH PARITY
220	Teletype No. 1, Input	20
221	Teletype No. 1, Output	A1
222	Teletype No. 2, Input	A2
223	Teletype No. 2, Output	23
224	Teletype No. 3, Input	A4
225	Teletype No. 3, Output	25
226	Teletype No. 4, Input	26
227	Teletype No. 4, Output	A7
228	Teletype No. 5, Input	A8
229	Teletype No. 5, Output	29
22A	Teletype No. 6, Input	2A
22B	Teletype No. 6, Output	AB
	.	
	.	
	Etc.	
	.	
23E	Teletype No. 16, Input	3E
23F	Teletype No. 16, Output	BF

INTERRUPT STATION ASSIGNMENTS

Teletypes may be connected to any interrupt mask station in groups of 4. The standard 3300 connection is:

Teletypes No. 1 - No. 4		Station 0 ,	A register mask = (01) HEX
Teletypes No. 5 - No. 8		Station 0 ,	A register mask = (01) HEX
	or	Station 1 ,	A register mask = (02) HEX
Teletypes No. 9 - No. 12		Station 0 ,	A register mask = (01) HEX
	or	Station 2 ,	A register mask = (04) HEX

Etc.

IX. CONTROL CONSOLE OPERATING PROCEDURES

The control console is located on the front panel of the CPU. The control panel is pictured in Figure 4. The switches and indicators are described in Table 2.

A. Turn-On Procedures

To turn computer on, set the power switch on the rear panel of the computer to the "ON" position. (All peripheral devices should be turned on after the computer is turned on and turned off before the computer is turned off.) The CLEAR button on the computer console should then be pushed.

B. Display Contents of Single Memory Location

The following procedure is used to display the contents of a single memory location:

1. Depress ENTER mode switch.
2. Set data bit switches to binary number of desired page.
3. Press B selector button (sets high order byte of BC counter).
4. Set data bit switches to binary number of location in page.
5. Press C selector button (sets low order byte of BC counter).
6. Depress DISPLAY mode switch.
7. Press CORE selector button.
8. Read contents of memory in MEMORY display section of panel.

C. Read Consecutive Memory Locations

To read a series of successive memory locations without having to set the location every time, simply follow the single location procedure described for the first byte desired. Then simply continue pressing the CORE button which will cause the program counter to increment by one each time the button is depressed. Remember, when using this method however, that the location shown by the BC display is the address of the *next* logical location and not the address of the byte seen in the MEMORY display.

D. Modify the Contents of a Single Memory Location

The procedure required to insert or modify data in a single location is as follows:

1. Follow procedure for setting BC counter to desired address (Steps 1 through 5 of Section B).
2. Depress ENTER mode switch (should already be depressed).
3. Set data entry switches to desired bit configuration.
4. Press CORE button (this enters data into address).

E. Modify the Contents of Consecutive Memory Locations

Perform all steps in Section D for first location. Repeat steps 3 and 4 for each byte to be entered. The program counter will automatically be stepped each time the CORE button is depressed.

F. Enter or Modify Data in Registers

The following procedure is used to enter or modify data in the registers:

1. Depress ENTER mode switch.
2. Set data bit switches to desired configuration.
3. Press selector button of register desired;

   Z for Z-Register
   A for A-Register
   S for Status Register
   M for M-Register
   B for B-Register
   C for C-Register

G. Single Step Program Execution

A stored program may be examined in detail by executing it step by step in the following manner:

1. Set BC counter to address of first instruction to be executed.
2. Depress RUN mode switch.
3. Press STEP button. This will execute one complete instruction.
   The BC counter will show address of next instruction and the register will show their present contents.
4. Continue to press STEP button for each desired instruction execution until program end is reached.

H. Automatic Program Execution

To run a loaded program:

1. Set the B and C registers to the desired starting address.
2. Depress the RUN mode switch.
3. Press the CLEAR button.
4. Press GO button. If program is correct, it should run until a halt instruction is encountered.
5. To stop program, press the STEP button. This will halt the program.

I. Execute Command from Control Panel

To execute command from the console, use the following steps:

1. Depress ENTER switch.
2. Set the data bit switches to the value of the 2nd byte of the instruction.
3. Press the M button. This will enter the 2nd byte of the instruction into the M-register.
4. Set the data bit switches to the value of the 1st instruction byte.
5. Press EXQ button. This will cause the instruction to be executed.

TABLE 2 – 3300 COMPUTER CONSOLE SWITCHES AND INDICATORS

ITEM	TYPE		FUNCTION
Clear Button	**Clear**	**Push Button**	**Initializes Computer and I/O Logic**
Z-Register Display	Z		Shows by light display the contents of the Z-Register.
A-Register Display	A	Bit	Shows by light display the contents of the Z-Register.
Status Reg. Display	S	Configuration	Shows by light display the contents of the Status Register.
B Counter Display	B	Display Lamps	Shows by light display the high-order byte of the program counter. (Memory page of next instruction)
C Counter Display	C		Shows by light display the low-order byte of the program counter. (Address within page of next instruction)
Memory Display	Memory		Shows by light display the last byte of data fetched from memory.
I/O Light	I/O	On/Off	Is lit when I/O bus is busy.
Carry Light	CA	Display Lamp	Is lit when last carry producing instruction resulted in a carry. (Status Register Bit 5)
Load Button	Load	Push	Loads first block of program from peripheral device (Enter Mode). (If hardware bootstrap option is installed)
Step Button	Step	Buttons	Steps thru memory one consecutive address at a time. (To single step thru program execution) or stops program execution.
Execute Button	EXQ		Executes a command loaded in data bit switches and the M-register, (Enter Mode).
GO Button	GO		Causes a program to automatically run until a halt (Run Mode).
Run Mode Switch	Run	Interconnected	When depressed, the machine is in Run Mode (Execute Instructions).
Display Mode Switch	Display	Push Switches	When depressed, the machine is in Display Mode (Display memory or registers).
Enter Mode Switch	Enter		When depressed, the machine is in Enter Mode (Enter data into memory or registers).
Data Bit Entry Switches	80,40,20,10 8,4,2,1	On/Off Push Switches	Used to set bit configuration for data entry. (In position = 1, out position = 0.)
Selector Buttons	B, C, Z, A, S, M, Core	Push Buttons	Enter or display data into or from corresponding location when pressed. (Into B, C counters, A-Register, Z-Register, S-Register, M-Register or core.) (From Core)

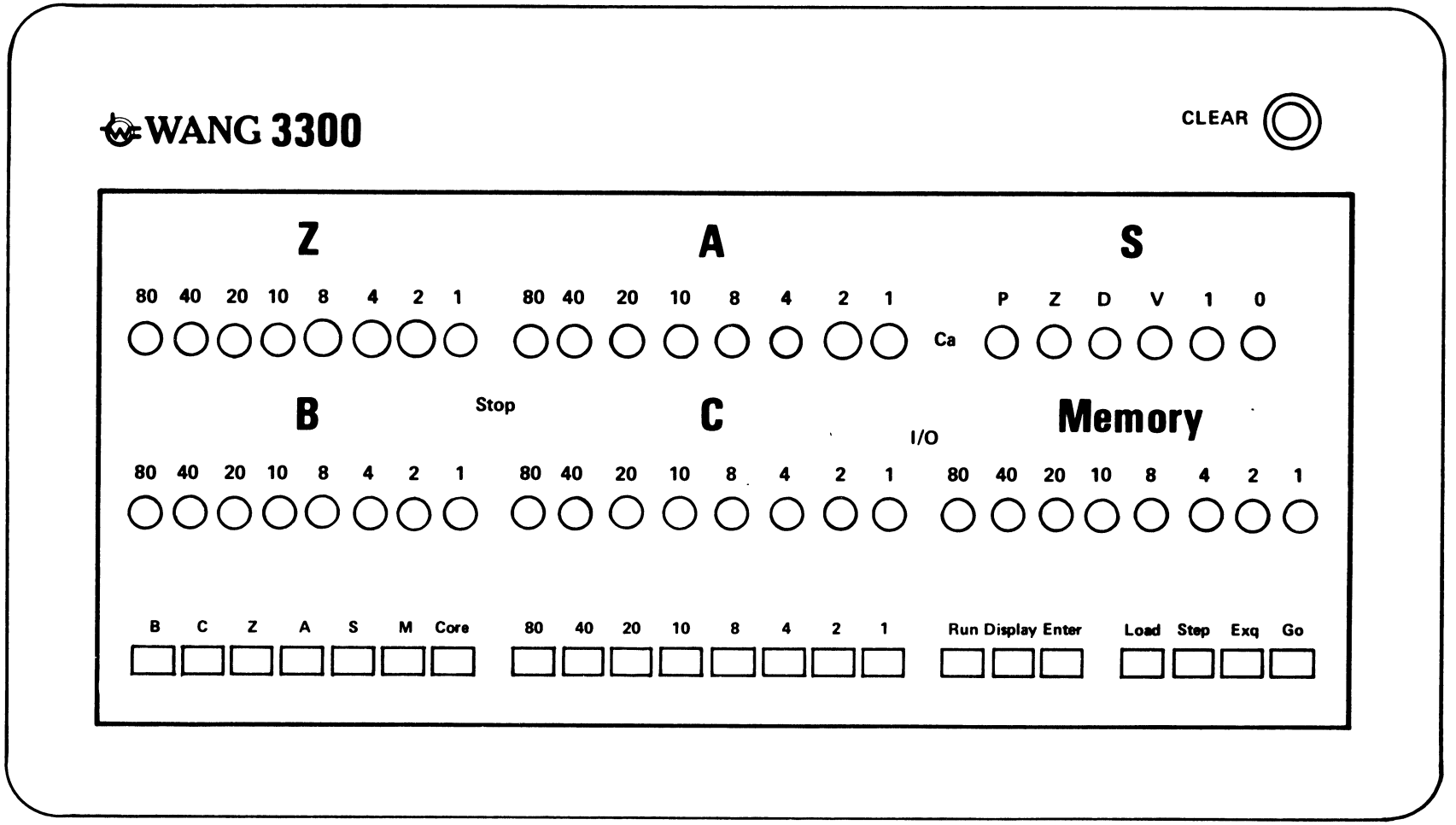


FIG. 4. WANG 3300 CONSOLE

(This page intentionally left blank.)

APPENDIX A

LIST OF INSTRUCTIONS BY GROUP

APPENDIX A – LIST OF INSTRUCTIONS BY GROUP

MNEMONIC NAME		HEXIDECIMAL GP CODE									INSTRUCTION FORMAT	PAGE WHERE DESCRIBED
			A	* A	C	* C	+ A	–A	+ C	–C		
		NO MEMORY REFERENCE	ABSOLUTE PAGE DIRECT	ABSOLUTE PAGE INDIRECT	CURRENT PAGE DIRECT	CURRENT PAGE INDIRECT	ABSOLUTE PAGE INDIRECT, AUTO-INCR.	ABSOLUTE PAGE INDIRECT, AUTO-DECR.	CURRENT PAGE INDIRECT, AUTO-INCR.	CURRENT PAGE INDIRECT, AUTO-DECR.		
A. MEMORY REFERENCE GROUP												
BA	BOOLEAN AND		48	49	4A	4B					1	12
BO	BOOLEAN OR		58	59	5A	5B					1	12
BX	BOOLEAN EXCLUSIVE OR		50	51	52	53					1	13
INC	INCREMENT		40	41	42	43					1	13
JEI	JUMP, AND ENABLE INTERRUPT		54	55	56	57					1	13
JMP	JMP		44	45	46	47					1	13
JST	JUMP AND STORE LOCATION		4C	4D	4E	4F					1	13
B. MEMORY REFERENCE GROUP WITH AUTO–INDEX												
AC	ADD WITH CARRY		88	89	8A	8B	8C	8D	8E	8F	2	14
ADD	ADD		80	81	82	83	84	85	86	87	2	14
AMC	ADD TO MEMORY WITH CARRY		98	99	9A	9B	9C	9D	9E	9F	2	14
C	COMPARE		90	91	92	93	94	95	96	97	2	14
DAM	DOUBLE ADD TO MEMORY		A8	A9	AA	AB	AC	AD	AE	AF	2	14
DCM	DOUBLE COMPARE TO MEMORY		A0	A1	A2	A3	A4	A5	A6	A7	2	14
DL	DOUBLE LOAD		B0	B1	B2	B3	B4	B5	B6	B7	2	14
DU	DOUBLE UNLOAD		B8	B9	BA	BB	BC	BD	BE	BF	2	15
LA	LOAD A		C0	C1	C2	C3	C4	C5	C6	C7	2	15
LZ	LOAD Z		D0	D1	D2	D3	D4	D5	D6	D7	2	15
UA	UNLOAD A		C8	C9	CA	CB	CC	CD	CE	CF	2	15
UAH	UNLOAD A HIGH DIGIT		E8	E9	EA	EB	EC	ED	EE	EF	2	15
UZ	UNLOAD Z		D8	D9	DA	DB	DC	DD	DE	DF	2	15
XMA	EXCHANGE MEMORY AND A		E0	E1	E2	E3	E4	E5	E6	E7	2	15
C. IMMEDIATE REFERENCE GROUP												
AI	ADD IMMEDIATE	18									3	15
BAI	BOOLEAN AND IMMEDIATE	1D									3	15
BOI	BOOLEAN OR IMMEDIATE	1F									3	15
BXI	BOOLEAN EXCLUSIVE OR IMMEDIATE	1E									3	16
CI	COMPARE IMMEDIATE	19									3	16
DLI	DOUBLE LOAD IMMEDIATE	1C									3	16
LAI	LOAD A IMMEDIATE	1A									3	16
LZI	LOAD Z IMMEDIATE	1B									3	16

APPENDIX A, CONT' – LIST OF INSTRUCTIONS BY GROUP

MNEMONIC	NAME	HEXIDECIMAL OP CODE										INSTRUCTION FORMAT	PAGE WHERE DESCRIBED
			A	* A	C	* C	+ A	– A	+ C	– C			
		NO MEMORY REFERENCE	ABSOLUTE PAGE DIRECT	ABSOLUTE PAGE INDIRECT	CURRENT PAGE DIRECT	CURRENT PAGE INDIRECT	ABSOLUTE PAGE INDIRECT, AUTO-INCR.	ABSOLUTE PAGE INDIRECT, AUTO-DECR.	CURRENT PAGE INDIRECT, AUTO-INCR.	CURRENT PAGE INDIRECT, AUTO-DECR.			
D. SHIFT AND ROTATE GROUP													
RT	ROTATE A LEFT	22										5	16
RTC	ROTATE A LEFT WITH CARRY	23										5	16
SH	SHIFT A LEFT	20										5	17
SHC	SHIFT A LEFT WITH CARRY	21										5	17
SBJ	SHIFT BINARY DOUBLE & JUMP				08							6	17
SBCJ	SHIFT BIN. DBLE. W/CARRY, JUMP				09							6	17
SDJ	SHIFT DECIMAL DOUBLE & JUMP				0A							6	17
E. CONDITIONAL JUMP GROUP													
JEQ	JUMP IF EQUAL				27							6	17
JGT	JUMP IF GREATER THAN				24							6	17
JLT	JUMP IF LESS THAN				26							6	17
JNE	JUMP IF NOT EQUAL				25							6	18
F. CONDITIONAL SKIP GROUP													
SAA	SKIP IF ANY A	12										4	18
SMA	SKIP IF MIXED A	13										4	18
STA	SKIP IF TRUE A	10										4	18
SFA	SKIP IF FALSE A	11										4	18
STS	SKIP IF TRUE S	14										4	18
SFS	SKIP IF FALSE S	15										4	18
STI	SKIP IF TRUE I/O	30										4	18
SFI	SKIP IF FALSE I/O	31										4	19

APPENDIX A, CONT' – LIST OF INSTRUCTIONS BY GROUP

MNEMONIC	NAME	HEXIDECIMAL OP CODE										
			A	* A	C	* C	+ A	– A	+ C	– C		
		NO MEMORY REFERENCE	ABSOLUTE PAGE DIRECT	ABSOLUTE PAGE INDIRECT	CURRENT PAGE DIRECT	CURRENT PAGE INDIRECT	ABSOLUTE PAGE INDIRECT, AUTO-INCR.	ABSOLUTE PAGE INDIRECT, AUTO-DECR.	CURRENT PAGE INDIRECT, AUTO-INCR.	CURRENT PAGE INDIRECT, AUTO-DECR.	INSTRUCTION FORMAT	PAGE WHERE DESCRIBED
G.	**REGISTER TRANSFER AND MANIPULATION GROUP**											
AZAJ	ADD Z TO A AND JUMP				04						6	19
DNJ	DOUBLE NOT AND JUMP				03						6	19
HLTJ	HALT AND JUMP				00						6	19
JZA	JUMP VIA Z AND A	OF									6	19
LZAJ	LOAD FROM Z AND A & JUMP				0B						6	19
NJ	NOT A AND JUMP				02						6	20
PARJ	PARITY AND JUMP				01						6	20
TASJ	TRANSFER A TO S AND JUMP				0E						6	20
TSAJ	TRANSFER S TO A AND JUMP				06						6	20
TZAJ	TRANSFER Z TO A AND JUMP				07						6	20
XZAJ	EXCHANGE Z AND A & JUMP				05						6	20
OFS	OFF STATUS	0D									4	20
ONS	ON STATUS	0C									4	20
H.	**INPUT/OUTPUT, AND INTERRUPT GROUP**											
AKIJ	ACKNOWLEDGE INTERRUPT & JUMP				32						6	21
DSIJ	DISABLE INTERRUPT & JUMP				33						6	21
ONMJ	ON INTERRUPT MASK & JUMP				39						6	21
CIOJ	CONTROL SIGNAL TO I/O & JUMP				3C						6	21
RDDJ	READ DATA & JUMP				3A						6	21
WRDJ	WRITE DATA & JUMP				3B						6	21
TIAJ	TRANSFER I/O TO A & JUMP				38						6	21

APPENDIX B

ALPHABETIC LIST OF INSTRUCTIONS

APPENDIX B – ALPHABETIC LIST OF INSTRUCTIONS

MNEMONIC NAME		HEXIDECIMAL OP CODE										
			A	* A	C	* C	+ A	– A	+ C	– C		
		NO MEMORY REFERENCE	ABSOLUTE PAGE DIRECT	ABSOLUTE PAGE INDIRECT	CURRENT PAGE DIRECT	CURRENT PAGE INDIRECT	ABSOLUTE PAGE INDIRECT, AUTO-INCR.	ABSOLUTE PAGE INDIRECT, AUTO-DECR.	CURRENT PAGE INDIRECT, AUTO-INCR.	CURRENT PAGE INDIRECT, AUTO-DECR.	INSTRUCTION FORMAT	PAGE WHERE DESCRIBED
AC	ADD WITH CARRY		88	89	8A	8B	8C	8D	8E	8F	2	14
ADD	ADD		80	81	82	83	84	85	86	87	2	14
AI	ADD IMMEDIATE	18									3	15
AKIJ	ACKNOWLEDGE INTERRUPT & JUMP				32						6	21
AMC	ADD TO MEMORY WITH CARRY		98	99	9A	9B	9C	9D	9E	9F	2	14
AZAJ	ADD Z TO A AND JUMP				04						6	19
BA	BOOLEAN AND		48	49	4A	4B					1	12
BAI	BOOLEAN AND IMMEDIATE	1D									3	15
BO	BOOLEAN OR		58	59	5A	5B					1	12
BOI	BOOLEAN OR IMMEDIATE	1F									3	15
BX	BOOLEAN EXCLUSIVE OR		50	51	52	53					1	13
BXI	BOOLEAN EXCLUSIVE OR IMM.	1E									3	16
C	COMPARE		90	91	92	93	94	95	96	97	2	14
CI	COMPARE IMMEDIATE	19									3	16
CIOJ	CONTROL SIGNAL TO I/O & JUMP				3C						6	21
DAM	DOUBLE ADD TO MEMORY		A8	A9	AA	AB	AC	AD	AE	AF	2	14
DCM	DOUBLE COMPARE TO MEMORY		A0	A1	A2	A3	A4	A5	A6	A7	2	14
DL	DOUBLE LOAD		B0	B1	B2	B3	B4	B5	B6	B7	2	14
DLI	DOUBLE LOAD IMMEDIATE	1C									3	16
DNJ	DOUBLE NOT AND JUMP				03						6	19
DSIJ	DISABLE INTERRUPT & JUMP				33						6	21
DU	DOUBLE UNLOAD		B8	B9	BA	BB	BC	BD	BE	BF	2	15
HLTJ	HALT AND JUMP				00						6	19
INC	INCREMENT		40	41	42	43					1	13
JEI	JUMP AND ENABLE INTERRUPT		54	55	56	57					1	13
JEQ	JUMP IF EQUAL				27						6	17
JGT	JUMP IF GREATER THAN				24						6	17
JLT	JUMP IF LESS THAN				26						6	17
JMP	JUMP		44	45	46	47					1	13
JNE	JUMP IF NOT EQUAL				25						6	18
JST	JUMP AND STORE LOCATION		4C	4D	4E	4F					1	13
JZA	JUMP VIA Z AND A	0F									6	19
LA	LOAD A		C0	C1	C2	C3	C4	C5	C6	C7	2	15
LAI	LOAD A IMMEDIATE	1A									3	16

APPENDIX B, CONT' – ALPHABETIC LIST OF INSTRUCTIONS

		HEXIDECIMAL OP CODE										
			A	* A	C	* C	+ A	– A	+ C	– C		
		NO MEMORY REFERENCE	ABSOLUTE PAGE DIRECT	ABSOLUTE PAGE INDIRECT	CURRENT PAGE DIRECT	CURRENT PAGE INDIRECT	ABSOLUTE PAGE INDIRECT, AUTO-INCR.	ABSOLUTE PAGE INDIRECT, AUTO-DECR.	CURRENT PAGE INDIRECT, AUTO-INCR.	CURRENT PAGE INDIRECT, AUTO-DECR.	INSTRUCTION FORMAT	PAGE WHERE DESCRIBED
LZ	LOAD Z		D0	D1	D2	D3	D4	D5	D6	D7	2	15
LZAJ	LOAD FROM Z AND A & JUMP				0B						6	19
LZI	LOAD Z IMMEDIATE	1B									3	16
NJ	NOT A AND JUMP				02						6	20
OFS	OFF STATUS J	0D									4	20
ONMJ	ON INTERRUPT MASK & JUMP				39						6	21
ONS	ON STATUS	0C									4	20
PARJ	PARITY & JUMP				01						6	20
RDDJ	READ DATA AND JUMP				3A						6	21
RT	ROTATE A LEFT	22									5	16
RTC	ROTATE A LEFT WITH CARRY	23									5	16
SAA	SKIP IF ANY A	12									4	18
SBCJ	SHIFT BIN. DOUBLE W/CARRY, JUMP				09						6	17
SBJ	SHIFT BIN. DOUBLE & JUMP				08						6	17
SDJ	SHIFT DEC. DOUBLE & JUMP				0A						6	17
SFA	SKIP IF FALSE A	11									4	18
SFI	SKIP IF FALSE I/O	31									4	19
SFS	SKIP IF FALSE S	15									4	18
SH	SHIFT A LEFT	20									5	17
SHC	SHIFT A LEFT WITH CARRY	21									5	17
SMA	SKIP IF MIXED A	13									4	18
STA	SKIP IF TRUE A	10									4	18
STI	SKIP IF TRUE I/O	30									4	18
STS	SKIP IF TRUE S	14									4	20
TASJ	TRANSFER A TO S & JUMP				0E						6	21
TIAJ	TRANSFER I/O TO A & JUMP				38						6	20
TSAJ	TRANSFER S TO A & JUMP				06						6	20
TZAJ	TRANSFER Z TO A & JUMP				07						6	20
UA	UNLOAD A		C8	C9	CA	CB	CC	CD	CE	CF	2	15
UAH	UNLOAD A HIGH DIGIT		E8	E9	EA	EB	EC	ED	EE	EF	2	15
UZ	UNLOAD Z		D8	D9	DA	DB	DC	DD	DE	DF	2	15
WRDJ	WRITE DATA & JUMP				3B						6	21
XMA	EXCHANGE MEMORY AND A		E0	E1	E2	E3	E4	E5	E6	E7	2	15
XZAJ	EXCHANGE Z AND A & JUMP				05						6	20

(This page intentionally left blank.)

APPENDIX C

TABLE OF INSTRUCTIONS BY OP CODE

APPENDIX C – TABLE OF INSTRUCTIONS BY OP CODE

LOW ORDER HEX DIGIT OF OP CODE BYTE

HIGH ORDER HEX DIGIT OF OP CODE BYTE

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	HLTJ	PARJ	INJ	DNJ	AZAJ	XZAJ	TSAJ	TZAJ	SBJ	SBCJ	SDJ	LZAJ	ONS	OFS	TASJ	JZA
1	STA	SFA	SAA	SMA	STS	SFS			AI	CI	LAI	LZI	DLI	BAI	BXI	BOI
2	SH	SHC	RT	RTC	JGT	JNE	JLT	JEQ								
3	STI	SFI	AKIJ	DSIJ					TIAJ	ONMJ	RDDJ	WRDJ	CIOJ			
4	← INC →				← JMP →				← BA →				← JST →			
5	← BX →				← JEI →				← BO →							
6																
7																
8	← ADD →								← AC →							
9	← C →								← AMC →							
A	← DCM →								← DAM →							
B	← DL →								← DU →							
C	← LA →								← UA →							
D	← LZ →								← UZ →							
E	← XMA →								← UAH →							
F																

APPENDIX D

WANG 3300 ASCII CHARACTER CODE SET

APPENDIX D – WANG 3300 ASCII CHARACTER CODE SET

			7	0	0	0	0	1	1	1	1
			6	0	0	1	1	0	0	1	1
4	3	2	5 / 1	0	1	0	1	0	1	0	1
0	0	0	0	Null		(Space)	0	@	P		
0	0	0	1		X - ON	!	1	A	Q		
0	0	1	0			"	2	B	R		
0	0	1	1		X - OFF	#	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	Bell		' (Apos)	7	G	W		
1	0	0	0	Back Space		(	8	H	X		
1	0	0	1	Tab		)	9	I	Y		
1	0	1	0	Line Feed		*	:	J	Z		
1	0	1	1	Clear Tab	ESC	+	;	K	[		
1	1	0	0	SET TAB	⩽	, (Comma)	<	L	\		
1	1	0	1	CARRIAGE RETURN	≠	- (Minus)	=	M	]		
1	1	1	0	Upper Shift	⩾	.	>	N	↑		
1	1	1	1	Lower Shift	o (Degree)	/	?	O	←		

WANG
LABORATORIES, INC.

836 North Street
Tewksbury, Massachusetts 01876
TELEPHONE (617) 851-7311
TWX 710-343-6769
TELEX 94-7421

WANG EUROPE, S.A.

Rue Anatole France 115-121
1030 Brussels, Belgium
TELEPHONE 15.48.01/02/03

WANG ELECTRONICS LTD.

40-44 High Street
Northwood, Middlesex, England
TELEPHONE Northwood 27677

WANG LABORATORIES GMBH

Moselstrasse No. 4
Frankfurt 6000
West Germany
TELEPHONE (611) 23-00-49

WANG SKANDINAVISKA AB

Drottninggatan 80
4th Floor
Stockholm, Sweden
TELEPHONE 21.07.15

WANG LABORATORIES (CANADA) LTD.

180 Duncan Mill Road
Don Mills, Ontario
TELEPHONE (416) 449-7890
TELEX 06-21-7549

WANG LABORATORIES (TAIWAN) LTD.

9, Sec. 5, Nanking East Road
Taipei, Taiwan
TELEPHONE 77.96.19 or 71.35.78

PHI COMPUTER SERVICES, INC.

800 Massachusetts Avenue
Arlington, Massachusetts 02174
TELEPHONE (617) 648-8550

WANG MEDICAL SYSTEMS, INC.

836 North Street
Tewksbury, Massachusetts 01876
TELEPHONE (617) 851-7311
TWX 710-343-6769
TELEX 94-7421

WANG NEDERLAND N.V.

Damstraat 2
Utrecht, Netherlands
TELEPHONE 030-930947

WANG PACIFIC LTD.

61, King Yip Street, 1st floor
Kwun Tong, Kowloon
TELEPHONE K-444536

Printed in U.S.A.
700-0365
10-71-8C