TM 9-1220-221-10/1

TECHNICAL MANUAL

PROGRAMMING MANUAL FOR COMPUTER, GUN DIRECTION, M18 (FADAC)

HEADQUARTERS, DEPARTMENT OF THE ARMY

AUGUST 1972

TECHNICAL MANUAL
NO. 9—1220—221—10/1

HEADQUARTERS
DEPARTMENT OF THE ARMY
Washington, D.C., *4 August 1972*

PROGRAMMING MANUAL
COMPUTER, GUN DIRECTION, M18 (FADAC)

CHAPTER 1.	GENERAL DESCRIPTION	PAGE
1.1	Purpose	1-1
1.2	Physical and Operational Characteristics	1-1
1.3	Functional Description	1-4
1.4	Description of Memory	1-7
1.4.1	Program Addressable Special Registers	1-10
1.5	Control Buttons	1-12
1.6	Number Systems Used	1-13
CHAPTER 2.	WORD FORMAT AND INSTRUCTION REPERTOIRE	
2.1	Word Formats	2-1
2.2	Instruction Repertoire	2-6
2.2.1	Arithmetic	2-8
2.2.2	Store and Load	2-10
2.2.3	Transfer	2-17
2.2.4	Shift and Cycle	2-18
2.2.5	Logic and Special Instructions	2-22
2.2.6	Discrete Input/Output	2-26
2.2.7	Parallel Input/Output	2-34
2.2.8	Read Commands	2-36
2.2.9	BCD and Alpha-numeric Input	2-38
2.2.10	Write Commands	2-42
CHAPTER 3.	CODING FOR FADAC	
3.1	Minimum Access Coding	3-1
3.2	Optimizing Assignments - Main Memory	3-3

		PAGE
3.3	Optimizing Assignments - R&Q-Loops	3-7
3.4	Use of Registers for Temporary Storage	3-9
3.5	Use of R and Q-Loops	3-13
3.6	Instruction Modification	3-14
3.7	Store Instruction Restrictions	3-17
3.8	Linkage to Subroutines	3-18
3.9	Table Look-up	3-20
3.10	Memory II/Memory III Addressing Technique	3-22
3.11	Use of 146 NI	3-32
3.12	Word Input/Output	3-36
CHAPTER 4.	FADAC OPERATING PROCEDURES	
4.1	Control Panel	4-1
4.2	FADAC Field Table	4-8
4.3	Signal Data Reproducer (SDR)	4-8
4.4	Input	4-11
4.4.1	Octal Control Characters	4-13
4.4.2	Preparation of Paper Tapes	4-15
4.5	Preparation of the FADAC System	4-16
4.5.1	Starting the System	4-17
4.6	Input Through External Control	4-17
4.6.1	Paper Tape (SDR)	4-17
4.6.2	Paper Tape (Mechanical Reader)	4-19
4.7	Input Through Program Control	4-22
APPENDICES		
A	Powers of Two Table	A-1
B	Octal-Decimal Integer Conversion Table	B-1

		PAGE
C	Numeric Input/Output Codes	C-1
D	Alpha-numeric Input/Output Codes	D-1
E	Optimization Table	E-1
F	Summary of FADAC Instruction Repertoire	F-1

LIST OF FIGURES

FIGURE		PAGE
1.1	Functional Diagram of System	1-5
1.2	Conceptual Layout of FADAC Memory	1-8
2.1	FADAC Octal Format	2-2
2.2	Instruction Word Format	2-2
2.3	Data Word Formats in BCD and Alpha-numeric Codes	2-6
2.4	Shift and Cycle Command Summary	2-20
2.5	DIA Pin and Line Connections	2-28
2.6	Matrix Position Codes	2-29
2.7	DIA Command-Binary Information	2-29
2.8	Discrete Output Command Summary	2-34
2.9	Input/Output Command Format	2-36
3.1	Multiplexing Diagram	3-2
4.1	Production Engineering FADAC	4-2
4.2	Relationship Between Nixie Lights and D-Register	4-7
4.3	Signal Data Reproducer (SDR)	4-9

LIST OF FIGURES (Cont'd)

FIGURE		PAGE
4.4	Block Diagram of a Typical FADAC System	4-12
4.5	Mechanical Reader	4-20

LIST OF TABLES

TABLE		PAGE
1.1	Summary of Register Addresses	1-12
1.2	Control Button Address	1-13

CHAPTER 1

GENERAL DESCRIPTION

1.1 PURPOSE

The purpose of this publication is to describe techniques required to program the Computer, Gun Direction, M18 (FADAC).

1.2 PHYSICAL AND OPERATIONAL CHARACTERISTICS

FADAC is a general purpose, transistorized digital computer designed for field use. By proper programming, it may be used to perform a variety of tasks. Information flow is serial by bit, parallel by function, allowing up to 12,800 operations per second.

Size.

24 inches by 14 inches by 34 inches.

Weight.

Approximately 210 lbs.

Power.

Three-phase, 4 wire, 400 Hz system; 120 volts line to neutral/208 volts line to line, approximately 750 watts.

Temperature.

-25°F to 125°F (external ambient at sea level); with rear cover installed, to -40°F. Automatic temperature protection is provided.

Commands.

One command per word; each command contains both address of operand and address of next instruction (1 + 1 system).

Numbers.

Straight binary for internal operations; automatic conversion to other codes for input-output; two's complement notation for negative numbers.

Word Length.

Thirty-two programmable bits; sign bit and 31 magnitude bits for numerical value.

Memory Type.

Magnetic Disc, 6000 rpm nominal speed of rotation.

Storage Capacity.

Sixty-four channels of 128 words each (8192 words) in main memory. Also provided are two 16 word high-speed loops for rapid access, three 1-word registers for arithmetic operations and control, and one 2-word register for output display information storage. All channels and loops have one read and one write head, except the 16-word loops which have an additional read head.

Pulse Repetition Rate.

Nominal pulse repetition rate, 460 kilopulses per second.

Access Times.

One word-time is 78 to 83 micro-seconds. One disc revolution requires approximately 10 milliseconds.

Input.

Input to the computer is from the manual keyboard or the mechanical tape reader on the control panel, at approximately 10 characters/second, or from external sources, such as: Reproducer, Signal Data, AN/GSQ-64, a high speed photo electric reader designated the Signal Data Reproducer (SDR) (600 characters/sec); another FADAC; magnetic tape (4000 char/sec max); other devices. Input may be in 5-channel Teletype or 8-channel Fieldata code.

Output.

FADAC can transmit data to either the Nixie tube visual display unit on the control panel, or through the output plug, to any output device such as a printer, paper tape punch unit, another FADAC, or other Fieldata or Teletype equipment. Output information is in 5-level teletype, 8 level Fieldata or 2-wire Teletype codes, as required. In addition to the standard codes, FADAC can be programmed to generate any 8-bit code. Output rates depend on the output code selected, as well as the output equipment. Maximum rates are as follows:

(1) 4000 char/sec in A/6

(2) 3550 char/sec in A/5 and A/4

(3) 650 char/sec in octal or BCD

Additional Features of the Computer.

(1) Parity check on information transfers.

(2) Verify indicator (located on the SDR) on input in Program-Fill mode.

(3) Gating for external support equipment to permit logic failure isolation to single printed circuit board.

(4) Marginal test circuit forpreventive maintenance.

(5) Voltage transient warning indicator.

(6) Temperature warning indicator.

(7) High speed (2 bits-at-a-time) multiplication, division, and shifts.

(8) Self-checking for overflow.

(9) Standardized printed circuit boards to minimize logistic problems.

1.3 FUNCTIONAL DESCRIPTION

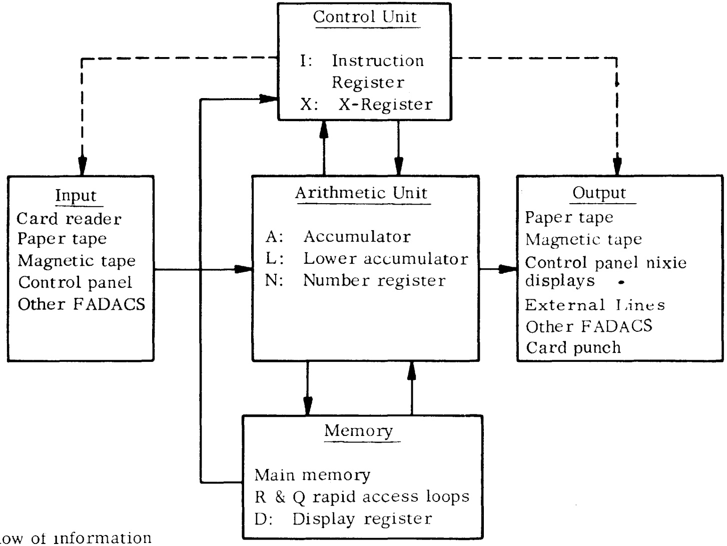
A functional diagram of the FADAC system is shown in Figure 1.1.

Control Unit.

The computer control unit interprets and processes the programmed instructions. Each instruction word called from memory is processed by the control logic through two non-program-addressable registers, X and I. During execution of the current instruction in the I-register, the next instruction is stored in the X-register. On completion of execution of the current instruction, the next instruction is sent to the I-register, and the new next instruction is read into the X-register. If the next instruction is found just as execution of the current instruction is completed, the X-register is bypassed, and the instruction is sent directly to the I-register. (See discussion of Multiplexing, Section 3.1.)

Arithmetic Unit.

The Arithmetic Unit performs the actual work involved in problem solution. In addition to the four arithmetic operations, this section can shift and cycle numbers right and left, and assist in operations which make it possible for the computer to make decisions, to accept and store input information, and to format and transmit output information. The Arithmetic Unit is



Arrows show flow of information

Figure 1.1. Functional Diagram of System

comprised of arithmetic logic, input/output logic, and three one-word, program-addressable registers: the Accumulator (A), the lower Accumulator (L), and the Number register (N).

The A-register is the principal arithmetic register in FADAC. It is employed in all arithmetic, logical, input/output, and decision making operations.

The L-register is an extension of the A-register in some operations (as in Multiply and Divide); and functions independently of A in others (as in Zero L, Store L). It also serves as the location counter for input/output operations.

The N-register holds the second operand for some operations (as in Multiply and Divide). It serves as an input/output buffer, and is used in program control transfer operations to store the transfer instruction, if that instruction is flagged.

The A, L, and N-registers are physically part of the FADAC memory disc.

Memory Unit.

The FADAC memory unit stores information in the form of instruction words and data words. It is comprised of 8192 words of main memory, plus two 16-word rapid access loops (R and Q) and one 2-word display loop (D).

Input.

Data and instructions can be placed in FADAC memory using any of the devices listed in Section 1.2. The Signal Data Reproducer (SDR) is the primary input device.

Output.

FADAC output is transmitted to the NIXIE display panel, or through the FADAC plug (J-10) to any external device, as listed in Section 1.2.

1.4 DESCRIPTION OF MEMORY

General.

FADAC memory is a rotating disc, coated with ferrous oxide similar to the coating on conventional magnetic tape. The disc rotates at approximately 6000 rpm under stationary read and write heads. To record information on the disc, an electrical signal generates a magnetic field in a write head. This field produces magnetization on the disc which remains, even when the computer is de-energized, until replaced by new information. The read heads are similar to the write heads, but operate in reverse. A change in magnetization on the disc produces an electrical current in the head. These signals produce the 1's and 0's for the binary operation of the computer.

Memory Layout.

A conceptual layout of the FADAC memory is shown in Figure 1.2. Although the actual position of the heads and the relative positions of the tracks are not exactly as shown, this diagram provides a useful overall pictorial diagram. The disk is designed in channels, each equipped with read and write heads. The sector track is permanently recorded and has a read head only. The purpose of the sector track is to supply data which notifies the control unit which sector is about to pass under the read heads for the other channels. The main memory read heads are all at the same effective sector location at any given point in time.

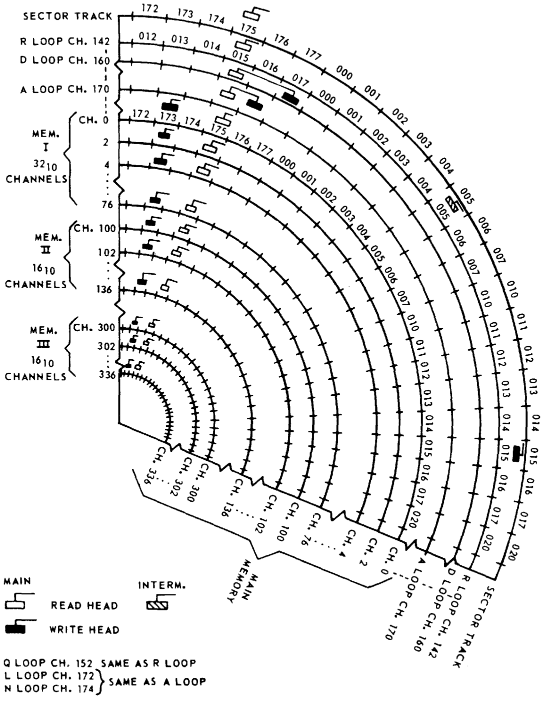


Figure 1.2. Conceptual Layout of FADAC Memory

Sector Track Contents.

The sector track contains 128 words of factory recorded information, available serially from a read flip-flop. A binary number appears in every word of the sector track. In each successive word the number is increased by 1, such that the octal numbers 0 to 177 are represented. These numbers serve as reference for numbering the memory sectors. Also recorded on the sector tract are the Teletype (TT) and Fieldata (FD) tape codes, in inverted format, for each of the 16 BCD and Octal characters recognized by FADAC. The four least significant bits of each sector number represent the BCD and Octal translation of the TT and FD tape code for a given character. The tape codes are repeated eight times around the sector track. See Section 2.2.7 for a description of how this information is used during I/O operations and Appendix C for a list of BCD and OCTAL codes.

Main Memory.

The 64 main memory channels are addressed by the even octal numbers 0 through 136 and 300 through 336. The 128 words or sectors of each channel are addressed by consecutive octal numbers 0 through 177 inclusive. To clarify discussion, the channels in main memory will be referenced throughout this manual as follows:

Memory I	= Channels	0 through 76
Memory II	= Channels	100 through 136
Memory III	= Channels	300 through 336

Hot and Cold Memory.

Main memory is further divided into Working Storage (Hot Memory) in which both the read and write heads are energized, and Permanent Storage (Cold Memory) in which the read heads ONLY are energized. At programmer's option, the size of working storage may be specified as either 4, 12, or 16 channels, and the FADAC "Hot Channel Select Switch"(see Section 4.5) manually set to energize the write heads of specific Hot Channels, as listed below:

Working Storage Size	Hot Channels Set (In Octal)
4	70 - 76
12	70 - 76 110 - 116 130 - 136
16	40 - 76

With the SDR connected to FADAC and the Auxiliary Memory Switch on the SDR in the OFF position, all write heads in main memory are energized to allow all of memory to be filled. However, during normal field use, memory is filled, the SDR disconnected, and only the contents of Hot memory can be varied. This design was adopted to assure that the contents of permanent memory are not inadvertently destroyed because of operator error or power failure during field use.

1.4.1 PROGRAM-ADDRESSABLE SPECIAL REGISTERS

Description.

The A, L, N, D, R, and Q-registers are short, recirculating loops. Each bit is sensed by the read head and rewritten by the write head. Each word is shifted, bit-by-bit, through the loop. The loop information remains the same unless changed by some arithmetic or control operation. A, L, and N are 1-word loops whose contents are rewritten each word time. D is a 2-word loop, the contents of which are rewritten two sectors ahead of the read head. Both the R and Q-loops have an intermediate read head located in the middle of the loop. This head is not involved in the recirculation process. However, it is used automatically by FADAC to expedite execution of a command which causes the contents of a sector in the R or Q-loop to be read.

The R and Q-Loops.

The rapid access loops, R and Q, may have octal sector addresses 0 through 177. However, only the last four bits, (00 through 17) are recognized. When addressing the R and Q-loops, therefore, only sectors 00 through 17 should be used. Note the contents of each sector of the R and the Q-loops are repeated eight times around the R and Q-loop tracks. The channel address of the R-loop is 142.00 and of the Q-loop, 152.00.

One-Word Loops.

The A, L, and N-registers each effectively occupy a full track on the FADAC memory disc. The contents of each loop are always immediately available under a read head. They are addressed as 170.00, 172.00 and 174.00 respectively.

The D-Register.

When the main memory read heads are over any even numbered sector, the contents of D.0 are under a read head; while when the main memory read heads are at any odd numbered sector, the contents of D.1 are available, since the D-register also effectively occupies a full track on the memory disc. Since the read and write heads of the D-loop are spaced two sectors apart both heads are effectively at the same sector of D at any given word time. The D-loop address is 160.000 (D.0) and 160.001 (D.1).

Summary of Register Address.

All register addresses are summarized in Table 1.1.

TABLE 1.1 SUMMARY OF REGISTER ADDRESSES

Register	Channel Address (Octal)	Sector Address (Octal)	Description
R	142	000 - 017	16 words
Q	152	000 -017	16 words
D.0	160	000 or even sec.	1 word
D.1	160	001 or odd sec.	1 word
A	170	000	A-register (Accumulator)
L	172	000	L-register (Lower Accumulator)
N	174	000	N-register (Number Accumulator)

1.5 CONTROL BUTTONS

The eight control buttons located on the front panel of FADAC are effectively wired directly to the memory locations shown in Table 1.2.

TABLE 1.2

CONTROL BUTTON

ADDRESS

Control Buttons	Address (Octal)
Sample Matrix	000 . 000
Test	000 . 001
Set Up	000 . 002
Recall	000 . 003
Send	000 . 004
Compute	000 . 005
Trig	000 . 006
Receive	000 . 007

When a control button is depressed, the flip-flop is set for Memory II, FADAC enters the compute mode, and the instruction stored in the control button location is executed. Computation is usually initiated in this manner. Note that the control button locations may be called in a program and their contents executed under program control.

1.6 NUMBER SYSTEMS USED

Number systems applicable to the FADAC computer are the decimal system, the octal system and the binary system. Information going into or coming from the computer is either decimal, octal or alphabetic. However, the computer operates internally with binary numbers only, and all negative numbers are represented in two's complement notation.

CHAPTER 2

WORD FORMATS AND INSTRUCTION REPERTOIRE

2.1 WORD FORMATS

General.

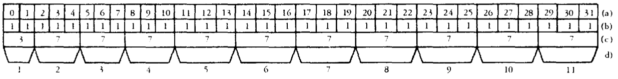
Information stored in FADAC memory contains no inherent distinction between numerical data and computer instructions. Both are stored as 36-bit binary coded numbers, of which 32 bits are programmable, and 4 a.e machine-generated. Of the 4 non-programmable bits, 1 is the parity bit, and 3 are used for synchronization. When a word is read from memory, the state of the computer logic will determine whether the information is to be interpreted as a data word or an instruction.

FADAC Octal Formats.

Each 32 bit FADAC word may be represented by 11 FADAC Octal digits. The first digit is formed from the two most significant bits of the binary word, and has a maximum value of 3. Each of the remaining digits is formed from three bits of the FADAC word, and has a maximum value of 7. Because of the short (i.e., two bits only) first digit in the FADAC format, a pure octal number and its FADAC-Octal representation are not the same.

Data Words in FADAC-Octal.

Information used as an operand is interpreted by the computer as a signed, 31 bit, binary-coded word, with a fixed point between the sign (bit position O) and the most significant bit. (See Fig. 2.1.) Positive numbers have a zero sign bit, and negative numbers a one. Negative numbers are represented in two's complement form.


0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 (a)
1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 (b)
3 7 7 7 7 7 7 7 7 7 7 (c)
d)
1 2 3 4 5 6 7 8 9 10 11

a) Bit Position

b) Max. Binary Value

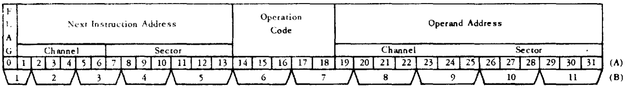
c) Max. Octal Value

d) FADAC Octal Position

Figure 2.1. FADAC Octal Format

Instruction Word Format in FADAC-Octal.

Commands are stored in the memory in the form of one instruction per computer word, represented by 11 FADAC-Octal digits. Each command consists of four separate fields; a flag, a next instruction address, an operation code (op code), and an operand address. (See Fig. 2.2.)


FLAG
Next Instruction Address
Channel
Sector
Operation Code
Operand Address
Channel
Sector
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 (A)
1 2 3 4 5 6 7 8 9 10 11 (B)

(A) Bit Position (B) FADAC-Octal Position

Figure 2.2. Instruction Word Format

The flag, which is in the sign position, provides for the modification of a FADAC instruction. The use of the flag is described for each instruction in the FADAC instruction list.

The Next Instruction Address, which tells the computer where to go next in the program sequence, consists of a channel and a sector portion. The channel portion consists of 7 bits. These are represented by 3 octal digits, which are evenly numbered from 000 to 174. The sector portion consists of 7 bits. These are represented by 3 octal digits which are numbered sequentially from 000 to 177.

The Operation Code, which represents the operation to be performed, usually consists of 6 bits. These are represented by 2 octal digits which are evenly numbered from 00 to 76. The shift, cycle, I/O, and most special instructions require a special format which is described in the sections covering these commands.

The Operand Address is the address of the number to be operated upon. It consists of a channel and sector portion and is in the same form as the Next Instruction Address.

A total of 15 octal digits seems to be required to represent an instruction word: 1 for the flag, 2 for the operation code, and 6 each for the addresses. Since a FADAC word is limited to only 11 digits, the four extra digits are combined in the following manner:

Let:

x_1 represent the octal digits in each of the four fields, and y_1 represent the FADAC Octal digits:

FADAC Instruction Word

Flag	Next Instruction		Op. Code	Operand Address		
	Channel	Sector		Channel	Sector	
x_1	$x_2\ x_3\ x_4$	$x_5\ x_6\ x_7$	$x_8\ x_9$	$x_{10}\ x_{11}\ x_{12}$	$x_{13}\ x_{14}\ x_{15}$	Octal
y_1	$y_2\ y_3$	$y_4\ y_5$	$y_6\ y_7$	$y_8\ y_9$	$y_{10}\ y_{11}$	FADAC Octal

Where:

$\underline{y_1 = x_1 + x_2}$ $\qquad$ $\underline{y_7 = x_9 + x_{10}}$

$y_2 = x_3$ $\qquad$ $y_8 = x_{11}$

$\underline{y_3 = x_4 + x_5}$ $\qquad$ $\underline{y_9 = x_{12} + x_{13}}$

$y_4 = x_6$ $\qquad$ $y_{10} = x_{14}$

$y_5 = x_7$ $\qquad$ $y_{11} = x_{15}$

$y_6 = x_8$

Note that all channels and operation codes are evenly numbered. Thus whenever an odd number ends a channel address field, a sector greater than or equal to 100 is called (i.e., $y_3 = x_4 + x_5$ or $y_9 = x_{12} + x_{13}$). Whenever an odd number ends the op code field, the channel of the operand address is $\geq$ 100 (i.e., $y_7 = x_9 + x_{10}$).

The first digit is derived as follows:

$$y_1 = x_1 + x_2$$

$y_1 = 0$ when no flag is called and NI channel is < 100

$y_1 = 1$ when no flag is called and NI channel is ≥ 100

$y_1 = 2$ when a flag is called and NI channel is < 100

$y_1 = 3$ when a flag is called and NI channel is ≥ 100

To illustrate: Given the instruction "Clear and Add (24) the contents of channel 102, sector 133, and go to channel 112, sector 123," the word format is as follows:

	Next Instruction		Op.	Operand Address		
Flag	Channel	Sector	Code	Channel	Sector	
0	1 1 2	1 2 3	2 4 (CLA)	1 0 2	1 3 3	Octal
1	1 3	2 3	2 5	0 3	3 3	FADAC-Octal

The FADAC octal word for this command is;

11323250333

Or consider the coding required for the instructions "Subtract* (02) the contents of channel 112, sector 113, and go to channel 112, sector 024."

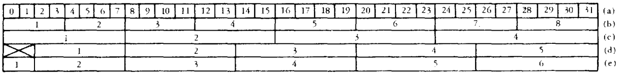
	Next Instruction			Operand Address		
Flag	Channel	Sector	Code	Channel	Sector	
2	1 1 2	0 2 4	0 2 (SUB*)	1 1 2	1 1 3	Octal
3	1 2	2 4	0 3	1 3	1 3	FADAC-Octal

The FADAC octal word for this command is:

31224031313

Input/Output Code Formats:

In addition to Octal data words, information may be input and/or output in BCD, Alpha-4, Alpha-5, or Alpha-6 code. Fig. 2-3 illustrates the data word format for each of these codes.



0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 (a)
1 2 3 4 5 6 7 8 (b)
1 2 3 4 (c)
1 2 3 4 5 (d)
1 2 3 4 5 6 (e)

(a) Bit Position

(b) BCD Code =4 bits/character; 8 characters/ FADAC word

(c) Alpha-4 Code = 8 bits/character; 4 characters/ FADAC word (First bit of each character is inverted on output)

(d) Alpha-5 Code = 6 bits/character; 5 characters/ FADAC word (Bits 0 and 1 are ignored)

(e) Alpha-6 Code = 6 bits/character; 6 characters/ FADAC word (4 most significant bits of first character are stripped on input, machine-generated on output)

Figure 2.3. Data Word Formats in BCD and Alpha-numeric Codes

2.2 INSTRUCTION REPERTOIRE

General.

All FADAC instructions fall into seven general categories according to function:

Arithmetic
Store and Load
Transfer
Shift & Cycle
Logic and Special
Discrete Input/Output
Parallel Input/Output

Execution time for each instruction is shown in the Word Time portion of the description. Since one word time is approximately 78 to 83 microseconds, timing may be obtained by multiplying 80 x 10^{-6} by the number of word times required to execute each instruction.

In the instruction descriptions, the following conventions are used:

(1) M represents the operand address of the instruction being executed.

(2) (M)represents the contents of location M.

(3) C represents the channel address of the operand address, or the 2 least significant digits of the 4 digit extended op codes required for shift, cycle, I/O and most special commands.

(4) S represents the sector address of the operand address in all categories except in instructions requiring a special format.

(5) A flagged instruction is represented in mnemonic form by writing an asterisk following the operation code, and in machine code by placing a 1 in bit position 0. For example: CLA, CLA*, STO, STO*, TRA, TRA*. The effect of the flag is specified in the description of each instruction.

(6) Unless explicitly mentioned under the description of each command, special registers are

not affected by the execution of a given command.

(7) All numbers are Octal unless otherwise specified, except numbers used to identify bit positions, which are designated in decimal.

2.2.1 ARTHMETIC

General.

If overflow occurs upon execution of the following instructions, the computer will automatically flash the Error light indicator on the control panel. The next instruction will be recognized only if it is a TRANSFER ON OVERFLOW (TOV) instruction, in which case the Error light indicator will be turned off. If any other instruction follows, the computer will halt and the Error light will continue to flash.

Rules for the Sign in Arithmetic Operations.

The rules for the sign of an arithmetic operation are the same as those of algebra. A negative zero is never generated by an FADAC arithmetic operation. However, the number 20000000000, which is not interpreted as a negative zero, may be input to FADAC and operated upon.

<u>ADD</u> 00 ADD M Word Time = 1

The algebraic sum of the contents of M and the contents of the A-register replace the contents of the A-register. Overflow can occur. A flag bit of 1 will cause (M) to be put into the N-register. A flag bit of 0 leaves the contents of the N-register unaltered.

<u>SUBTRACT</u> 02 SUB M Word Time = 1

The word in location M is subtracted from the A-register, and the difference replaces the contents of the A-register. Overflow can occur. A flag bit of 1 will cause the contents of M to be put into the N-register. A flag bit of 0 leaves the contents of the N-register unaltered.

<u>MULTIPLY</u> 20 MPY M Word Time = 22

The contents of the memory location M are multiplied by the contents of the A-register. The product appears in the A and L-registers unrounded. The most significant 31 bits and the sign are in the A-register. The least significant 31 bits are in the L-register. Because the number in the L-register is the least significant part of the total product, it does not have a sign associated with it. The most significant bit of the 31 bits of the least significant part of the product appears in the sign position of the L-register. The least significant bit position of the L-register always contains a 0 after multiplication, which is not considered part of the product. Note that the contents of location M are stored in the N-register. The flag bit is ignored during execution of this instruction.

<u>DIVIDE</u> 30 DIV M Word Time = 22

The 63 bit number appearing in the A and L-registers is divided by (M). The most significant bit of the portion of the dividend that appears in the L-register is in the sign position. The least significant bit of the L-register is not considered part of the dividend and must be a zero. The quotient in the A-register will be affected if the value of this bit is 1.

If the flag bit is 0, the A-register will contain a rounded quotient and the L-register will contain a remainder whose value has no relationship to the rounded quotient.

If the flag bit is 1, the A-register will contain an unrounded quotient and the L-register will contain a remainder with the same sign as the divisor. This remainder satisfies the relation:

(Divisor × Quotient) + Remainder = Dividend

The contents of location M are put into the N-register. Overflow can occur if the absolute value of the contents of the A-register is equal to or greater than the value of the contents of location M.

CLEAR AND ADD 24 CLA M Word Time = 1

The contents of location M are transferred to the A-register. Location M can refer to any addressable location. A flag bit of 1 transfers the previous contents of the A-register into the L-register. If the flag bit is 0, the contents of the L-register remain unaltered.

CLEAR AND SUBSTRACT 26 CLS M Word Time = 1

The negative of the contents of memory location M (in 2's complement form), is transferred to the A-register. Location M can refer to any addressable location. A flag bit of 1 transfers the previous contents of the A-register into the L-register. If the flag bit is 0, the contents of the L-register remain unaltered.

2.2.2 STORE & LOAD

Most store instructions include a capability to write the specified information into the N-register as into location M. If M is in Hot memory, the indicated change occurs in M and in the special register. However, if M is in Cold memory, only the N-register is changed. Each instruction description includes information as to whether a flag is required to cause storage into N. Note particularly the special precautions required when an STO or STP command is executed.

STORE A-REGISTER 50 STA M Word Time = 1

The contents of the A-register are stored in location M. Location M can refer to any addressable location. If the flag bit is 1, the word in A will be put in the N-register as well as into location M.

NOTE: If M refers to registers A, L or N:

With flag bit = 0, no storage takes place into M or N.

With flag bit = 1, no storage takes place into M, but the contents of A are sent to N.

STORE L 52 STL M Word Time = 1

The contents of the L-register are stored in location M. M may be any addressable location. If the flag bit is one, the word in L will be put into the N-register as well as into location M.

NOTE: If M refers to registers A, L or N:

With flag bit = 0, no storage takes place into M or N.

With flag bit = 1, no storage takes place into M, but the contents of L are sent to N.

STORE N-REGISTER 40 STN M Word Time = 1

The contents of the N-register are stored in location M. M may be any addressable location. However, if M refers to the A, L or N-register, no storage takes place, and the instruction serves only as a one-word time delay. The flag bit is ignored in this instruction.

STORE D 42 STD M Word Time = 1

The contents of sector i of the 2-word D-register are stored in location M, where i = 00 if M is even, and i = 01 if M is odd. M may be any addressable location. If the flag bit is one, the contents of D.i will be stored in the N-register as well as in M.

NOTE: If M refers to registers A, L, or N:

With flag bit = 0, no storage takes place into M or N

With flag bit = 1, no storage takes place into M, but the contents of the D-loop under read head at time of execution are stored in N.

STORE OPERAND ADDRESS 70 STO M Word Time = 1 if M refers to R or Q.

Word Time = 3 if M refers to any other location.

(a) The operand address of the word in location M is replaced by the operand address of the word in the A-register (bits 19 through 31). The modified contents of M are stored in the N-register. M may be any addressable location.

(b) If M is located in Cold memory and no external input device, such as the SDR, is connected, the contents of M are always unchanged. However, the N-register is modified to contain the instruction in M, with its operand address replaced by the operand address currently in the A-register.

(c) If M is located in Memory II or III, and this memory is Hot, the sign bit of the L-register will determine whether (M) is modified by an STO or an STP instruction. If L is negative, no modification of (M) will occur, although the modified instruction will be stored in N. If L is positive, both (M) and (N) will be modified by execution of an STO or STP instruction. Therefore, to assure the modification of (M) in a Hot Memory II or Memory III location, the STO or STP could be preceded by a ZEL (Zero L) instruction. If M is in Memory

I, the sign bit in L hàs no effect on the execution of STO or STP commands.

(d) NOTE: If M refers to the A-register, A remains unchanged and the contents of A are stored in the N-register. If M refers to the L-register, the contents of L remain unchanged. However, the N-register is modified to contain the instruction in L, with its operand address (bits 19 thru 31) replaced by the operand address currently in A. If M refers to the N-register, bits 19 thru 31 in A are stored in bits 19 thru 31 of N. The remaining portion of the N-register is unchanged. The flag bit is ignored in this instruction.

<u>STORE PROGRAM ADDRESS</u>	60 STP M	Word Time = 1 if M refers to R or Q.
		Word Time = 3 if M refers to any other location.

(a) The next instruction address of the word in location M is replaced by the next instruction address of the word in the A-register (bit positions 1 thru 13). The modified contents of M are stored in the N-register. M may refer to any addressable location.

(b) If M is located in Cold memory and the SDR is disconnected, the contents of M remain unchanged. However, the N-register is modified to contain the instruction in M, with its NI address replaced by the NI address currently in the A-register (bits 1 thru 13).

(c) See Part (c) under Store Operand Address. The contents of the L-register have the same affect on an STP instruction as on an STO.

(d) If M refers to the A-register, A remains unchanged and the contents of A are stored in N. If M refers to the

L-register, L remains unchanged. However, the N-register is modified to contain the instruction in L, with its NI address replaced by the NI address currently in the A-register (bits 1 thru 13). If M refers to the N-register bits 1 thru 13 are stored in bits 1 thru 13 of N. The remaining portion of N remains unchanged. The flag bit is ignored in this instruction.

<u>STORE R</u>   62 STR M   Word Time = 22

The contents of the 20 words of the R-loop are transferred through the D-register to memory starting at location M. The flag bit is ignored. If M refers to a main memory location, each word transferred will be taken from that sector of the R-loop whose address agrees with the last four bits of the sector address into which the word is to be stored. For example, when an STR instruction with an operand sector address of 074 is executed, the first word stored will be taken from sector 14 of the R-loop. Successive words will be taken from sectors 15, 16, 17, 0, 1...13 of the R-loop and stored in sectors 075, 076, 077, 100, 101...113 of the main memory channel specified in M. The D-loop will contain the last two words transferred; that is, D.0 will hold the contents of the last even numbered location transferred, and D.1 the contents of the last odd numbered location transferred. In the example given above, D.0 will hold the new contents of sector 112, and D.1 will hold the new contents of sector 113.

If M refers to a main memory channel, and its sector is greater than 160, the word transmitted from R.0 will be stored in sector zero of the channel in which M is located.

Example: Given STR 20.166. Resultant storage will be:

Contents of R.06 $\longrightarrow$ 20.166

R.07 $\longrightarrow$ 20.167

. .

. .

. .

R.16 $\longrightarrow$ 20.176

R.17 $\longrightarrow$ 20.177

R.00 $\longrightarrow$ 20.0

. .

. .

. .

R.04 $\longrightarrow$ 20.04

R.05 $\longrightarrow$ 20.05

If M refers to a location in the Q or R-loop, the contents of R will go into Q or R, displaced by two sectors. For example, if a STR instruction is given which has an operand address of Q-3, sectors 3, 4, 5, 6...2 of R will be transferred to sectors 5, 6, 7, 0...4 of Q. At the end of the execution, D.0 will contain the new contents Q.4, and D.1 the new contents of Q.3. Similarly, if a STR instruction is given which has an operand address of R.6, sectors 6, 7, 10, 11...5 of R will be transferred to sectors 10, 11, 12, 13...7 of R. At the end of the execution, D.0 and D.1 will hold the new contents of sectors R.6 and R.7 respectively.

If M refers to the D-loop, the first word transferred appears in the sector of D specified and the last word transferred in the remaining sector of D. If the operand address is D.0, the first word with an even numbered address to pass under the read head of the

R-loop after the Instruction Read phase will be the first word transferred. For example, if channel 00, sector 55 contains the instruction STR D.0, the word in R.16 will appear in D.0 and the word in R.15 will appear in D.1 at the end of the execution phase. If the instruction were STR D.1, the contents of R.17 and R.16 would appear in D.1 and D.0 respectively.

<u>LOAD Q-REGISTER</u>	72 LDQ M	Word Time = 20
<u>LOAD R-REGISTER</u>	72 LDR* M	Word Time = 20

The contents of 20 consecutive words of memory starting with location M are transferred to the specified rapid access loop. The specific loop, i.e., R or Q, is determined by the flag bit. A flag bit of 1 indicates the R-loop, while a flag bit of 0 indicates the Q-loop.

Each word transferred to Q or R is stored in the sector corresponding to the last four bits of the sector from which the word was transferred. For example, when an LDQ instruction with an operand sector of 074 is executed, the first word transferred will go into sector 14 of the Q-loop. Successive words will go intosectors 15, 16, 17, 0, 1, 2, ...13. If M refers to a main memory channel, and the sector is greater than 160, the word transmitted from sector 177 will be followed by the word from sector 0 of the same channel.

The A, L, and N-registers and the D-loop are unaffected by the execution of these commands.

If M = A, L, or N, the contents of the specified register will replace the contents of all 20 sectors of the Q or R-loop. Similarly, if M = D.i, the contents of D.0 will go into all the even numbered sectors of the Q or R-loop, while the contents of D.1 will replace the contents of all the odd numbered sectors. If the operand of a LDR* command is in the R-loop, or if the operand of a LDQ command is in the Q-loop, the contents of the referenced loop remain unchanged.

2.2.3 TRANSFER

In each of the following instructions, a flag bit of 1 will cause the transfer instruction to be placed in the N-register if and only if the transfer to M is executed. A flag bit of 0 will leave the contents of the N-register unchanged.

TRANSFER UNCONDITIONALLY
14 TRA M Word Time = 1

Control is unconditionally transferred to the instruction found in location M. The NI is ignored in this instruction, except when the NI contains a 146 channel address. (See Section 3.10.)

TRANSFER ON PLUS 10 TPL M Word Time = 1

If the contents of the A-register are positive($\geq$ zero), control is transferred to the instruction in location M; if not, control is transferred to the address specified in the NI portion of the transfer command.

TRANSFER ON ZERO 12 TZE M Word Time = 1

If the contents of the A-register equal zero, control is transferred to location M; if not, control is transferred to the address specified in the NI portion of the transfer command.

TRANSFER ON OVERFLOW 16 TOV M Word Time = 1

If overflow occurs, control is transferred to the instruction in location M. If not, control is transferred to the address specified in the NI portion of the transfer command. In either case, the Error light indicator on the control panel will not flash on execution of this command.

Note that if an instruction which results in overflow is not immediately followed by a TOV command, the Error light will flash, and all computation will halt.

2.2.4 SHIFT & CYCLE

General.

In the following instructions the operand sector address portion of the instruction, designated by S, does not refer to a location in memory but to the number of binary positions the word in the A-register, or in the A and L-registers, is to be shifted or cycled. All shift and cycle op codes are made up of four octal digits, with the 2 least significant digits in the positions normally occupied by the channel portion of the operand address.

Overflow.

A change in the value of the sign bit in the A-register is normally recognized as overflow. However, overflow caused by a right cycle command is always ignored by FADAC, and the next program instruction is executed. Overflow can never occur with a right shift command. The flag bit of all right shift and right cycle instructions MUST be 0. Overflow caused by a left shift or cycle command will be ignored if the instruction flag bit is 0. However, a flag bit of 1 with a left shift or cycle command will cause the computer to check for overflow. If overflow occurs, the Error light indicator on the control panel will flash. The next instruction will be executed only if it is a TRANSFER ON OVERFLOW (TOV) command, in which case the Error indicator will stop flashing. If any other instruction follows, the computer will halt, and the

Error indicator light will continue to flash.

CAUTION: The contents of the shifted register(s) are not predictable when overflow occurs since the overflow condition may terminate the command before it is completed.

Execution Time.

The execution time for all shift and cycle instructions is given by the following relationship:

Convert S to decimal S'. The number of word times equals the integer portion only of

$$1/2\ (K + S' + 1)$$

where K = 0 for odd S' and where K = 1 for zero and even S'. This equation was derived from the following relationship:

$$\text{Word Time} = 1/2\ (1 + S' + \cos^2 \pi\, S'/2)$$

All shift and cycle commands are summarized in Fig. 2.4.

Accumulator Right Cycle 7600 ARC S

The contents of the A-register are shifted right the number of binary positions designated by S. The bits shifted off the right end of the A-register enter the left end of the A-register, through the sign position, in the same sequence.

Accumulator Right Shift 7602 ARS S

On execution of this command, the contents of the A-register are shifted right the number of binary places designated by S. The sign is spread in the left-hand bits; while bits shifted beyond the right-hand limits of the A-register are lost.

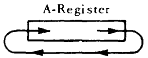
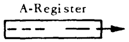
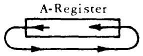
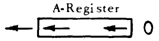
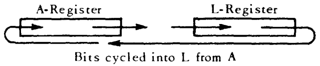
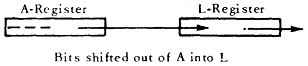
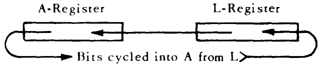
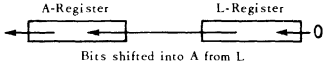
Command		Description		Flag
ARC 7600	Bits Re-enter left end of A when cycled off right end		Bits cycled right	Must be 0
ARS 7602	Sign is spread		Bits shifted out of A are lost	Must be 0
ALC 7604	Bits cycled left		Bits Re-enter right end of A when cycled off left end	May be 1 or 0
ALS 7606	Bits shifted out of A are lost		Zeroes fill vacated bit positions	May be 1 or 0
LRC 7620	Bits cycled out of L Re-enter A		Bits cycled right	Must be 0
LRS 7622	Sign is spread		Bits shifted out of L are lost	Must be 0
LLC 7624	Bits cycled left		Bits cycled out of A re-enter L	May be 1 or 0
LLS 7626	Bits shifted out of A are lost		Zeroes fill vacated bit positions	May be 1 or 0

Figure 2.4. Shift and Cycle Command Summary

Accumulator Left Cycle **7604 ALC S**

The contents of the A-register are shifted left through the sign bit position the number of binary places indicated by S. The bits shifted off the left end of the A-register enter the right end, in the same sequence.

Accumulator Left Shift **7606 ALS S**

The contents of the A-register are shifted left through the sign bit the number of binary places indicated by S. Positions left vacant as the contents of the A-register are shifted are filled with zeros. Any bits shifted beyond the sign position are lost.

Long Right Cycle **7620 LRC S**

The contents of the A and L-registers are shifted right the number of binary places designated by S. The bits shifted off the right end of the A-register are shifted into the L-register through its sign position. The bits shifted off the right end of the L-register are shifted into the left end of the A-register through its sign position.

Long Right Shift **7622 LRS S**

The contents of the A and L-registers are shifted right the number of binary places designated by S. The bits shifted off the right end of the A-register are shifted into the L-register through its sign bit position. The bits shifted off the right end of the L-register are lost. The sign of the A-register is spread.

Long Left Cycle **7624 LLC S**

The contents of the A and L-registers are shifted left the number of binary places indicated by S. The bits shifted left through the sign of the A-register are shifted into the right end of the L-register. Bits shifted through the sign position of the L-register are shifted into the right end of the A-register.

Long Left Shift 7626 LLS S

The contents of the A and L-registers are shifted left the number of binary places designated by S. The bits shifted through the sign position of the L-register are shifted into the right end of the A-register. Bits shifted beyond the sign position of the A-register are lost. The bits of the L-register vacated in the shifting process are filled with zeros.

2.2.5 LOGIC AND SPECIAL INSTRUCTIONS

General.

With the exception of the EXT, EQS and GES commands, all Logic and Special instructions are specified by a four digit op-code, with the two least significant digits in the positions normally occupied by the channel portion of the operand address. The sector portion of the operand address is ignored in these commands.

EXTRACT 34 EXT M Word Time = 1

The contents of the A-register are replaced by the logical AND of the contents of A and the contents of M. In other words, when both the bit in (M) and the corresponding bit in the A-register are 1, a one will appear in the same bit position of the A-register at the end of the execution. In all other cases, a zero will appear in that bit position of the A-register upon execution of this command. Location M can refer to any addressable location. The flag bit is ignored in this instruction.

TAKE ABSOLUTE VALUE 3770 ABS Word Time = 1

The contents of the A-register are replaced by their absolute value. If the contents are negative, they will be replaced by their corresponding positive value; while if the contents are positive they will remain unchanged. If the flag bit is zero, no further storage occurs. If the flag bit is one and the contents of A are negative, this negative value will be transferred to the L-register. If the flag bit is one and the contents of A are positive, L will remain unchanged. The operand sector is ignored in this command.

REPLACE A ON
MINUS FROM L 3772 RML Word Time = 1

If the contents of the A-register are negative and the flag bit of this instruction is zero, the complement of the contents of the L-register is sent to A, and L is unchanged. If the contents of the A-register are negative and the instruction flag bit is one, the complement of the contents of L is sent to A, and the original contents of A are sent to L. If the contents of the A-register are positive, A and L are unchanged regardless of the value of the flag bit. The sign of the original contents of the L-register in no way affects the execution of this command.

REPLACE A ON
MINUS FROM N 3774 RMN Word Time = 1

If the contents of the A-register are negative and the flag bit of this instruction is zero, the complement of the contents of the N-register is sent to A. The L and N-registers are unchanged. If the contents of the A-register are negative and the instruction flag bit is one, the complement of the contents of the N-register is sent to A; the original contents of A are sent to L; and N is unchanged. If the contents of the A-register are positive, A, L and N are unchanged. The sign of the original contents of the N-register in no way affects the execution of this command.

EQUAL SEARCH 64 EQS M Word Time = i + 2
(i = 0 to 176)

Starting with i = 0, the contents of the A-register are compared with those in memory location (M + i), in bit positions indicated by a mask appearing in the L-register. This mask must have been stored previously in the L-register by the program. It directs which bits of the A-register and memory should be compared. Wherever a 1 appears in the L-register, the corresponding bits in the A-register and location (M + i) will be checked.

If in all the designated bit positions, the contents of the A-register are found to be equal to the contents of location (M + i), the contents of location (M + i + 1) replace the contents of the A-register, and control is transferred to the next instruction.

If the contents of the A-register in all the designated bit positions are not equal to the contents of location (M + i), then i is increased by one and the comparison is repeated. This comparison search continues until agreement has been reached, or until the contents of operand sector 176 have been compared. If no agreement is reached after comparison search through sector 176, then the contents of A remain unchanged, and control is transferred to the next instruction.

If a complete channel is to be searched, the starting operand sector should be 177. Then if agreement were reached, the first and last possible word that would be picked up would be in sectors 000 and 177 respectively. If the flag bit is a 1, and agreement is reached, the previous contents of the accumulator replace the contents of L. If agreement is not reached, the flag bit has no effect. The execution time is the number of sectors reached, i, plus 2.

GREATER THAN OR EQUAL SEARCH 66 GES M Word Time = i + 2 (i = 0 to 176)

The execution of this instruction is the same as EQS, except that the comparison is satisfied if the value of the bit positions of (M) which are compared is greater than or equal to the value of the corresponding bit positions of (A).

HALT 3720 HLT Word Time = 1

Program computation is halted. The flag bit, operand sector and next instruction address are ignored in this command.

ZERO-L-REGISTER 3762 ZEL Word Time = 1

The L-register is cleared to zero. The flag bit and operand sector are ignored in this instruction.

INITIATE DISPLAY MODE 3764 IDM Word Time = 1

This command will cause the contents of the D-loop to be displayed, in BCD format, in the NIXIE tubes located on the control panels and will also activate the Battery display NIXIE. The Battery NIXIE is not part of the D-register, but is a wired-in display of the Battery button (A, B, C, D, or E) depressed on the matrix panel. Note that a battery button must be depressed when an IDM command is sensed, or no display will occur. Once activated, the NIXIE display will continue until a HALT DISPLAY MODE (HDM), or a READ or WRITE command is sensed. Computation can continue while in the display mode. The flag bit and operand sector are ignored in this command.

See Section 4.1 for a full description of the NIXIE display.

HALT DISPLAY MODE 3766 HDM Word Time = 1

Visual display mode is halted. The flag bit and the operand sector address are ignored by this command. Successive IDM and HDM commands can be given at any time.

INITIATE COMPUTE MODE 3726 ICM Word Time = 1

This command will turn on the Compute light located on the front panel of FADAC. If the Compute light is already on, the command will introduce a one word-time delay. It has no other effect on a program.

HALT COMPUTE MODE 3724 HCM Word Time = 1

This command will turn off the Compute light located on the front panel of FADAC, but does NOT halt computation. It has no other effect on a program. After an HCM command is sensed, the Compute light can only be relit if:

(1) an ICM command is sensed; or

(2) FADAC is turned off, then turned on, and one of the eight control buttons is depressed.

2.2.6 DISCRETE INPUT/OUTPUT

Input.

A single command, DIA, is available for a variety of input configurations to FADAC. These are described below:

DISCRETE INPUT TO
ACCUMULATOR 3640 DIA Word Time = 1

The DIA command places the following information into the A-register:

a. In bit position 0: A timing signal, with a period of 22 milliseconds, which is generated by an oscillator in FADAC. The signal is a square wave which remains at zero volts (0 to sign bit) for 11 milliseconds, and at −6 volts (1 to sign bit) for 11 milliseconds. The signal is provided primarily as a synchronization signal for output by bit to a standard two-wire teletypewriter, but may be used as a real-time clock in any application.

b. In bit position 1: Normally set to zero, this bit is set to one whenever any one of the 8 control buttons (Sample Matrix, Test, Set-up, Recall, Send, Compute, Trig, or Receive) is depressed; and remains at one only as long as pressure remains on the control button.

c. In bit positions 2 - 18: Up to 17 signals on lines connecting a single, or many, external devices to FADAC through Input Plug J - 17 and Output Plug J - 10. Note that the DIA command places information on specified pins of both plugs into the A-register simultaneously. (Figure 2.5 shows which pins on the I/O plugs are connected to each bit position.)

d. In bit positions 19 - 23: A 5 position switch is available to the programmer through the FADAC control panel buttons labelled A through E, which are located on the right hand side of the 8 x 8 matrix. With one of these buttons depressed, a DIA command will place its identifying code into bits 19 through 23 of the A-register. (See figure 2.7.) (Note in the Cannon Program these buttons are referred to as the Battery Buttons. However, they are available for use in any application.)

e. In bit position 24: A 2 position switch is available to the programmer through the control panel buttons labelled (1) and (2) which are located immediately above the 5 position switch, buttons A to E. With Button (1) depressed, a DIA command will place a 1 in bit position 24 of the A-register; with Button (2) depressed, a 0 appears in bit position 24. (In the Cannon Program, these buttons are referred to as the Weapons Buttons.)

f. In bit positions 25 - 30: The 8 x 8 matrix on the FADAC control panel greatly extends the input

BIT POSITION	F-LINE	PIN DESIGNATION
2	F-30-I	m
3	F-29-I	k
4	F-28-I	J On Input Plug J - 17
5	F-27-I	i
6	F-26-I	h
7	F-25-I	g
8	F-24-I	r
9	F-23-I	q
10	F-22-I	p On Output Plug J - 10
11	F-21-I	n
12	F-20-I	m
13	F-19-I	k
14	F-18	MM
15	F-17	LL
16	F-16	KK On Input Plug J - 17
17	F-15	JJ
18	F-14	HH

Figure 2.5. DIA Command Pin and Line Connections - Bits 2 Thru 18

COL. ↓	IDENTIFYING CODES							
H	70	71	72	73	74	75	76	77
G	60	61	62	63	64	65	66	67
F	50	51	52	53	54	55	56	57
E	40	41	42	43	44	45	46	47
D	30	31	32	33	34	35	36	37
C	20	21	22	23	24	25	26	27
B	10	11	12	13	14	15	16	17
A	00	01	02	03	04	05	06	07
ROW→	1	2	3	4	5	6	7	8

Figure 2.6. Identifying Codes for Each FADAC Matrix Position

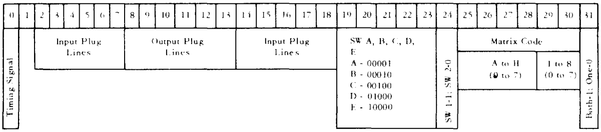


Figure 2.7. DIA COMMAND - Binary Information

f. In bit positions 25-30 (Continued)

capabilities of FADAC by providing a 64 position switch to the programmer. Each position may be programmed as a pointer to a separate element of a given complex program. With a row (labelled A, B, ..., or H) and column (labelled 1, 2, ..., or 8) button depressed, a DIA command will place the identifying code of the combination in bits 25 - 30 of the A-register. Figure 2.6 shows the 2-digit octal identifying code for each combination.

g. Bit 31: If only one button is depressed on the 8 x 8 matrix (a row or a column, but not both) a 0 will appear in bit position 31; if both are depressed, a 1 will fill bit position 31.

h. Figure 2-7. summarizes the information sent to the A-register by a DIA command.

The following example illustrates a use of the DIA command:

Example

The Problem.

Code an input routine for Matrix Position A - 1 through A -5; test if proper row and column button has been depressed; if so, transfer to one of 5 subroutines; if not, cause the No Solution Light to flicker, and halt. Assume that subroutines 1 through 5 begin in locations 10.00, 12.00, 14.00, 16.00 and 20.00, respectively, and that a transfer table has been stored which has the following octal format:

Octal Digit / Contents	1	2	3	4	5	6	7	8	9	10	11
	2	0	0	1	4	X	X	X	X	Y	Y

Explanation of the Transfer Table Format:

Digit 1: Set to 2 to indicate that a match has been found, and a control word picked up from the transfer table (i.e. - the word in A is negative).

Digit 2 and 3: Always 0

Digit 4 and 5: Set to 14, the TRA op code.

Digit 6 through 9(XXXX): Entrance location of each subroutine 1 through 5.

Digit 10 and 11 (YY): Matrix Codes 00 through 04 (A-1 through A-5).

TRANSFER TABLE

Location	Contents
0.172	00000000000
0.173	20014100001
0.174	20014120002
0.175	20014140003
0.176	20014160004
0.177	20014200000

Assume that when each selected subroutine has been executed, control is to return to the main program which starts at L_0 2.100 (00300).

INSTRUCTIONS

L_0	N_1	Op Code	Operand	Explanation
0.0	00201	36	4000	When sample matrix button is depressed, the program begins by executing DIA command(3640).
2.1	00202	76	0001	ARC 1: To place bit 31 in sign position.
2.2	00203	10	0403	TPL: if (+) go to Error Routine; if (–) continue.
2.3	00204	34	0604	EXT: Matrix code now in bit positions 26 to 31 in A-register. L_0 604 contains the Mask.

L 0	N 1	Op Code	Operand	Explanation
2.4	20205	24	0604	CLA*: Matrix code to L-register, Mask to A-register.
2.5	20206	25	7200	CLA*: Matrix code to A-register, Mask to L-register.
2.6	00207	64	0172	EQS: Search table stored in 0.172 through 0.177.
2.7	00210	10	0403	TPL: If +, wrong combo set in Matrix, go to error routine; if −, A-register now contains transfer instruction.
2.10	00213	76	0206	ARS 6: Position transfer instruction properly in A.
2.13	20300	15	7000	TRA* A and return to Main Program.
4.3	00404	37	1600	NSL-Error Routine - Wrong matrix position is set.
4.4	00000	37	2000	HLT - Halt, correct and start again.
6.4	00000	00	0077	Mask for EXT and EQS command

The DIA command may also be used to input any 2 to 16 bit code which is foreign to FADAC. This is accomplished by connecting up to 17 of the J10 and

J17 lines associated with the DIA command to any input device which generates the desired code. A timing signal generated by the external device must be connected to one of the 17 bits positions available. Duration of the timing signal must be long enough to allow input of each new character, and short enough to assure that each new character is sensed only once per input. With the period of the timing signal known, a program can be written to sample for this signal by a DIA, EXT, and TEST. When the timing signal is thus detected, the new code can be input by a second DIA command followed by an EXT (extract the number of bits which define a character), STA, and a delay designed to inhibit reinitiation of a loop back to test for the timing signal until the next character is available for input. An example of a code foreign to FADAC is USASCII, an 8 - bit code which can be input only in the manner described.

Discrete Output:

The eight discrete output commands set 3 flip-flops, OP1, OP2, and OP3, to any one of eight possible combinations. Each combination, in turn, either places a -6 volt signal on a given output line; or causes the No Solution Light (NSL) on the FADAC control panel to blink; or turns off the NSL or the signal on any output line (DOF). A description of each discrete output command is summarized in Figure 2.8.

Note that each Discrete Output command requires a four-digit operation code. The operand sector address is ignored in the execution of these commands. Also, note that the setting of the Discrete Output control flip-flops remains fixed until a new Discrete Output command is sensed. When FADAC is turned on, the output flip-flops are set to all zeros (effectively, a DOF command is executed).

Command Mnemonic	Octal Op-Code	OP-Flip-Flop Setting (in Binary)			OPL Line on which a -6 volt signal appears when the command is executed:	Pin to which given OPL line is connected on specified I/O Plug:	Function of Command
		Op3	Op2	Op1			
DOF	3700	0	0	0	None (All	None	Turns off the NSL or Sets all OPL lines to Zero
ODI	3702	1	1	1	OPL - 1	X of J-10	Sets OPL-1 to one
OD2	3704	0	1	0	OPL - 2	Y of J-10	Sets OPL-2 to one
OD3	3706	0	1	1	OPL - 3	Z of J-10	Sets OPL-3 to one
IDI	3710	1	0	0	OPL - 4	$\bar{a}$ of J-17	Sets OPL-4 to one
ID2	3712	1	0	1	OPL - 5	$\bar{b}$ of J-17	Sets OPL-5 to one
ID3	3714	1	1	0	OPL - 6	$\bar{c}$ of J-17	Sets OPL-6 to one
NSL	3716	1	1	1	None	None	Causes No Solution Light to blink

Figure 2.8. Discrete Output Command Summary

The function of the OD and ID commands is to to energize a given pin on the input or output plug, or de-energize all OPL line-connected pins on these plugs. They may be used to control external equipment by the presence or absence of -6 volts on a given line.

2.2.7 PARALLEL INPUT/OUTPUT

Introduction.

The parallel I/O commands cause FADAC to read information input through the keyboard, mechanical reader, magnetic tape, or any external device; or to write information out to another FADAC or to any of several external devices. The command structure allows the programmer to designate the I/O device; the kind of block (word or character); the block size; the I/O code (octal, BCD, or alpha-numeric); and, in all numeric WRITE commands, the code level (either 5-level Teletype (TT), or 8-level Fieldata (FD)). During input, the level is designated by the setting of the SDR (or SDR simulator) TT/FD switch. Parallel I/O commands are so named since each character enters FADAC on 5 (for TT) or 8 (for FD) information lines, (designated "I" lines) in parallel; and each character is output from FADAC on 5 or 8 data lines (designated "D" lines) in parallel. Input through the Mechanical Reader on the FADAC control panel is specified by the READ TAPE commands. Input through the MLU is specified by the READ EXTERNAL DEVICE commands. The flow of information through the special registers during I/O is a function of the particular I/O command executed, and is described in the following sections. In general, it is recommended that the contents of all special registers be saved prior to execution of any I/O command if they are of any significance in the balance of a given program. Note that the number of word times required to execute a parallel I/O command cannot be fully predicted.

The I/O Command Format:

Figure 2.9 illustrates the I/O command format. The flag bit is set to 0 to designate I/O by character, or to 1 to designate I/O by word. The value of the flag is added to the most significant digit of the NI address to form the first octal digit of the command word. The operation code occupies the 6th, 7th, 8th and 9th octal digits of the word. The block size is in digits 8, 9, 10 and 11. Thus, digits 8 and 9 are the sum of the two least significant digits of the operation code and the two most significant digits of the block size. The block size is designated as one less than the actual

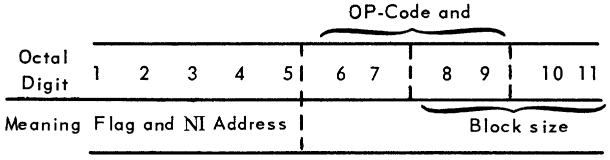


Figure 2.9. I/O Command Format

number of words or characters to be input or output. When the block is designated as a word, the maximum block size which may be specified per command is 1023 (1777). When I/O is by character, the maximum number of characters which can be put or output per command is a function of the number of characters which can be packed into a single FADAC word in the specified code.

2.2.8 READ COMMANDS

Octal Input:

Execution of a READ Octal command differs from all other parallel I/O in that the block type, block size and the Next Instruction portions of the command are ignored by FADAC; and control reverts to the input device. As each character is received, FADAC hardware inverts the tape code sensed, and searches the sector track (Section 1.4) for a match. The four least significant bits of each sector number represent the binary translation of the tape code for that number. When a match is found, the leading bit of the binary translation is tested. If it is a one, the character is an octal control character or a blank, and FADAC logic is set to perform the indicated operation (Halt, Compute, Fill, Verify, Enter, Location, Clear, or Blank). Both leading and non-leading blanks are ignored. If the leading bit is an O, it is stripped and the octal translation of the number is shifted into the right side of the A-register. If the tape code sensed is not found in the sector track, the character is ignored. If input is in Fieldata code, parity is checked. The Parity Light on the FADAC control panel will blink when a parity error is detected, halting input. Otherwise, input continues until a Compute or Halt code is received. The Compute code will cause control to be returned to FADAC, and the command contained in the address defined by the 14 least significant bits of the L-register will be executed. (See Appendix C for a list of Octal codes, and their TT and FD tape code equivalents.) The Halt code will stop all FADAC operations. If no Enter code is received before input is halted, the last 11 (or less) octal digits input will remain in the A-register. If more than 11 digits are received with no Enter code, the most significant digit is shifted out of the left side of A and lost, as each new digit enters the right side of the register. The contents of A, L, N, and all sectors of the Q-loop are destroyed by the input of no more than one full Octal word, while the contents of the D-loop and R-loop are unchanged. If more than one Octal word is to be input, the 5-digit octal address of the first location into which the data is to be stored must be entered right adjusted, followed by a location code. This causes the last 14 bits of A to replace the last 14 bits of the L-register. Input of an Enter code will send the operand address in L to the Q-loop and the contents of A to the N-register, a sector of the R-loop, both sectors of the D-loop and thence to the memory location last sent to the Q-loop from the L-register. L is incremented by 1 each time a word is stored. If several words are to be stored in contiguous memory locations, only the initial address must be forced into L, using the location code. To change the contents of L for non-contiguous storage, it is necessary to input the new 5-digit address, followed by a Location Code.

During octal input, the 5-digit location address may designate any part of FADAC memory. To address

Memory I, the first digit must be 0; to address Memory II, the first digit is 1; to address Memory III, the first digit is 3. When a Memory III (location address) is set in L, and an Enter code sensed, the memory control flip-flop (discussed in detail in Section 3.10) is set such that the information in A is stored in the designated Memory III location. Storage into Memory III will continue until a Memory II location address is sent to the L-register; at which time the memory control flip-flop will be reset to store inputs into Memory II. Designation of a Memory I location address will not change the current setting of the flip-flop. Storage into the designated Memory I location occurs regardless of this setting.

2.2.9 BCD AND ALPHA-NUMERIC INPUT

General.

BCD and alpha-numeric input is entirely under program and FADAC hardware control. When such input commands are executed, the flag bit is checked to determine block type (word or character), and the block size is saved in a temporary counter. The counter is automatically decremented by one each time a block is processed. Control is sent to the Next Instruction address when the counter equals zero, indicating input is completed.

Input by Character:

With the flag bit set to zero, each character shifted into A causes the block size counter to decrease by one. The characters remain in the A-register only, if the equivalent of one full word or less is input. If more than a full word (8 BCD characters, 5 Alpha-5, or 6 Alpha-6 characters) is received, the contents of A are loaded into the N-register, and subsequent characters are shifted into A. A maximum of two full FADAC words can be processed by character. No storage to main memory occurs automatically during character input. If the data received is to be stored, a Store A (STA) and Store N (STN) command must follow the input instruction. The original contents of A, L, and N are destroyed during character input, while the D, R, and Q-loops are unchanged.

Input by Word:

Prior to word input, the L-register must be programmed to contain the address of the first location into which the first word received is to be stored. Each time the A-register is filled with the number of characters which define a full word in the input code specified the address in L is sent to the Q-loop, and the contents of A are loaded into N. N is loaded into R; then written into the D-loop, and finally into the memory location specified in Q. L is incremented by one each time a word is written into main memory. The input cycle thru the registers is repeated until the indicated block size, plus one, has been input and stored.

BCD Input:

During the input of BCD codes, FADAC hardware inverts the tape code sensed, and searches the sector track for a match. If a match is found, the four least significant bits of the corresponding sector track number are the BCD translation of the tape code. These bits are shifted into the right hand end of the A-register. If no match is found, the character is ignored. Leading blanks are ignored, while imbedded blanks are processed. (See Appendix C for a list of BCD codes and their TT and FD tape code equivalents.)

BCD Input By Character:

Example: Since 8 BCD characters constitute one full FADAC word, a maximum of 16 BCD characters may be meaningfully input per command. To illustrate processing, consider the execution of a command requiring the input of 12 BCD characters. When input is finished, the contents of A and N will be as follows:

N-Register

Char	Char	Char	Char	Char	Char	Char	Char
1	2	3	4	5	6	7	8

A-Register

Char	Char	Char	Char	Char	Char	Char	Char
5	6	7	8	9	10	11	12

A contains characters 5 through 8 of the first word input plus characters 9 through 12---the last 4 characters entered. N contains characters 1 through 8 which constitutes the first full word received. Characters 1 through 4, which were in A when N was loaded, were shifted out of the left side of A when characters 9 through 12 entered the right side of A.

READ BCD Commands:

The following op-codes cause FADAC to set up the necessary hardware logic to accept BCD input from the indicated device:

Op-Code Mnemonic	Op-Code in FADAC Octal	Interpretation
RED	5500	Rd External Device in BCD
RTD	5520	Rd Tape (Mechanical Reader) Device in BCD
RKD	5560	Rd Keyboard in BCD

Alpha-Numeric Input:

Two alpha-numeric codes, designated Alpha-5 and Alpha-6, are processed by FADAC input logic. Alpha-5 code is defined as five 6-bit characters per word, or 30 information bits. The two additional bits which constitute a FADAC word have no significance. Alpha-6 code is defined as six 6-bit characters per word, with the 4 leading bits of the first character of each Alpha-6 shifted out of A and lost when the sixth character enters A. All alpha-numeric characters are initially received as an 8-bit Fieldata tape code. Parity is checked. If a parity error is detected, the Parity light on the FADAC control panel blinks until the RESET button is pressed. If Parity is good, the two leading bits of each tape code are stripped, and the 6 information bits are shifted into the right side of the A-register. No table look-up is required for these codes, since the 6 least significant bits of each FD tape code represents the binary configuration designated for a given character. (See Appendix D.) Therefore, I/O in any alpha-numeric code is much faster than is numeric code I/O. During character input, a maximum of 9 useful characters may be entered in Alpha-5 code, or 11 in Alpha-6 code.

READ Alpha-Numeric Commands:

The following op-codes cause FADAC to set up the necessary hardware logic to accept alpha-numeric input from the indicated devices:

Op Code Mnemonic	Op Code in FADAC Octal	Interpretation
RE5	5700	Rd External Device in Alpha-5
RT5	5720	Rd Tape Device in Alpha-5

Op Code Mnemonic	Op Code in FADAC Octal	Interpretation
RM5	5740	Rd Magnetic Tape in Alpha-5
RK5	5760	Rd Keyboard in Alpha-5
RE6	5600	Rd External Device in Alpha-6
RT6	5630	Rd Tape Device in Alpha-6
RM6	5640	Rd Magnetic Tape in Alpha-6
RK6	5660	Rd Keyboard in Alpha-6

2.2.10 WRITE COMMANDS

General.

All parallel output is entirely under program and FADAC hardware control. When such commands are executed, the flag bit is checked to determine block type, and the block size is saved in a temporary counter. The counter is automatically decremented by one each time a block is processed. Control is sent to the Next Instruction address when the block size, plus one, has been output.

Output by Character:

Prior to execution of a WRITE by character command, the characters to be output must be loaded, left justified, into the A-register. If more than one full word is to be output, the first X characters which define a full word in the specified output code must be sent to A, and the remaining characters must be left justified in N. The equivalent of two full FADAC words may be output by character (16 BCD characters, 10.Alpha-5, or 12 Alpha-6), except when the output code is Alpha-4. When one full word has been output, the contents of N are automatically loaded into A. Each character is shifted out of the left side of A, one character at a time until the designated block size, plus one, has been generated. The original contents of A, L and N are destroyed. The contents of all sectors of the R, Q and D-loops are unchanged.

Output by Word:

Prior to execution of a WRITE by word command, bits 18 to 31 of the L-register must be programmed to contain the address of the first word to be generated. During execution of a WRITE command, the contents of the address in L are written into the R-loop, the N-register, and the A-register, from which they are sent to the output lines, one character at a time. The address in L is sent to the Q-loop. L is incremented by one, and the cycle through the registers to the output lines is repeated until the number of words specified in the block size portion of the WRITE command, plus one, has been output. The contents of the D-loop are unchanged during output. When output is completed, the N-register contains the last word output; bits 18 to 31 of the L-register contain the last address from which data was output, plus 22. The Contents of A and all sectors of the R and Q-loop cannot be fully predicted.

Numeric Output:

During numeric output, in Octal or BCD code, each character shifted out of the left side of A is matched against the four least significant bits of the sector numbers inscribed on the sector track. Note that during look-up of octal characters (3 bits), the missing leading bit is interpreted as zero, since control characters are not stored in octal. Also, since the most significant octal digit of each word is composed of only 2 bits, both missing leading bits of that character are interpreted as zero. When a match is found, the corresponding tape code designated in the WRITE command (TT or FD) is inverted, sent to the D lines, and output. If output is in Octal code, FADAC hardware automatically generates an Enter code each time a full word is output, whether the block type be word or character. The Enter code must be counted as part of the designated block size when octal output is by character.

WRITE Numeric Commands:

The following op-codes cause FADAC hardware to set up the necessary logic to output data in Octal or BCD code to the indicated devices in either 5 level or 8 level tape code.

Op-Code Mnemonic	Op-Code in FADAC Octal	Interpretation
WEOT	4400	Write FADAC to External Device in Octal - TT
WFOT	4420	Write FADAC to FADAC in Octal - TT
WEOF	4440	Write FADAC to External Device in Octal - FD
WFOF	4460	Write FADAC to FADAC in Octal - FD
WEDT	4500	Write FADAC to External Device in Dec. - TT
WFDT	4520	Write FADAC to FADAC in Dec. - TT

Op-Code Mnemonic	Op-Code in FADAC Octal	Interpretation
WEDF	4540	Write FADAC to External Device in Dec. - FD
WFDF	4560	Write FADAC to FADAC in Dec. - FD

Alpha-Numeric Output:

Three alpha-numeric codes are output by FADAC, each of which are discussed below. Since no table look-up is required for alpha-numeric output, these codes are generated at a much faster rate than is possible during numeric output.

Alpha-4:

Alpha-4 is defined for output only. It is composed of 8 bits per character, 4 characters per FADAC word. Any 8-bit configuration pre-stored in FADAC may be generated by a Write Alpha-4 command. It is used to output a code foreign to FADAC, such as ASCII, or to control external devices by placing a control word on the D-lines. Output in Alpha-4 must be by character only, with a maximum block size of 3 specified (designating 4 characters actually to be generated). Prior to execution of a Write Alpha-4 command, the A-register must be loaded with the 4 (or less) characters to be generated. As each 8 bit character is shifted out of A, its most significant bit is inverted by FADAC hardware. The inverted first bit and the remaining 7 bits are then placed on the D-lines.

Alpha-5:

Alpha-5 is defined for both input and output as 5 six-bit characters per FADAC word. Unlike input, Alpha-5 output must be by word only. Each Alpha-5 word is automatically processed by shifting bits 0 and

1 out of the A-register, after which each six bit character is placed on the data lines, with a parity bit and a timing bit added to the most significant bit positions by FADAC hardware to form the 8-bit Fieldata code for that character.

Alpha-6:

During processing of a Write Alpha-6 command, the 2 leading bits of each Alpha-6 word are recognized as the 2 least significant bits of the first character. On output, these two bits are placed on D-lines 1 and 2, the next 4 bits are generated by FADAC, and a parity and timing bit added, to form an 8-bit Fieldata code on lines D-1 thru D-8. The value of the machine generated bits cannot be predicted or controlled. Therefore, the first character of each word output in Alpha-6 code may be meaningless. This characteristic limits the use of Alpha-6 code to communication between two FADAC's or between FADAC and a magnetic tape unit. In both applications, when each Alpha-6 word is read back into FADAC or the magnetic tape unit, the 6 leading bits of the first character are stripped, and only the two original bits are stored. The flow of information through the special registers is as described in the sections on "output by Character" and "Output by Word". Alpha-6 code may be output by word or character.

Write Alpha-Numeric Commands:

The following op-codes cause FADAC to set up the necessary hardware logic to allow output in the indicated Alpha-numeric codes:

Op-Code Mnemonic	Op-Code in FADAC Octal	Interpretation
WE4	4700	Write FADAC to External Device in Alpha-4 by Character
WF4	4720	Write FADAC to FADAC in Alpha-4 by character
WE5*	4700	Write FADAC to External Device in Alpha-5, by word
WF5*	4720	Write FADAC to FADAC in Alpha-5, by word
WE6	4600	Write FADAC to External Device in Alpha-6
WF6	4620	Write FADAC to FADAC in Alpha-6
WM6	4640	Write FADAC to Mag. Tape Alpha-6

* Note that the op-codes for the Write in Alpha-4 commands are identical to those for the Write in Alpha-5 commands. For the former, the flag bit must be 0, since output can be by character only; while for the latter, the flag bit must be 1 to designate output by word only.

CHAPTER 3

CODING FOR FADAC

3.1 MINIMUM ACCESS CODING

Multiplexing.

In the processing of all FADAC instructions, the following five phases are automatically completed:

(1) Instruction Search (IS)

(2) Instruction Read (IR)

(3) Operand Search (OS)

(4) Operand Read (OR)

(5) Execution (EX)

As illustrated in Figure 3.1, the phases are multiplexed; that is, performed simultaneously by FADAC hardware. Consider the multiplexing of five instructions, n − 1 through n + 3, each of which is assumed to be optimally coded, and each of which requires one word time to execute. Assume that n is the current instruction, n − 1 the immediately preceding instruction, and n + 1 through n + 3 the subsequent instructions. Let t = the word time of the execution of the preceding instruction, n − 1. In Figure 3.1, the multiplexing of the five instructions is shown from time t − 2 through time t + 2. At t − 2 (two word times before t) search is made for instruction n − 1. At time t − 1, instruction n − 1 is read, its operand is searched, and simultaneously, search is made for the current instruction, n. At time t, search is made for the next instruction, n + 1, instruction n is read and its operand searched; and at the same time the operand of the preceding instructions, n − 1, is read and instruction n − 1 is executed. At time t + 1,

Word Time				
t − 2	t − 1	t	t + 1	t + 2
IS_{n-1}	IS_n	IS_{n+1}	IS_{n+2}	IS_{n+3}
	IR_{n-1}	IR_n	IR_{n+1}	IR_{n+2}
	OS_{n-1}	OS_n	OS_{n+1}	OS_{n+2}
		OR_{n-1}	OR_n	OR_{n+1}
		EX_{n-1}	EX_n	EX_{n+1}

Figure 3.1. Multiplexing Diagram

the operand of the current instruction, n, is read, n is executed, and within the same word time, instruction search is made for n + 2, n + 1 is read and its operand searched, etc.

In all of the above it is assumed that each instruction requires only one word time to execute. If time to execute a command isgreater than one word time, then execution is initiated at the same time the current operand is read, and execution of the next command is delayed until execution of the current command is completed.

This multiplexing feature of FADAC hardware will decrease program execution time if the programmer places instructions and operands in memory such that:

1. Each instruction comes under a read head no later than the last word time of the execution phase of the previous instruction.

2. Each operand comes under a read head during the first word time following the completion of the previous instruction.

If these conditions are met, the program is optimally-coded, and there is no wait to read instructions or operands. The time lapse in fully executing an instruction is its execution time; the total elapsed time to complete a program is equal to the sum of the execution times of all instructions within the program.

Recall that the execution of most FADAC commands requires one word time, with the exception of the following:

Command	Number of Word Times Required for Execution
MPY	18
DIV	18
LDR*	16
LDQ	16
STR	18
All Shifts & Cycles	Variable
All Parallel I/O	Variable
STO	3 if M is in Main Memory
STP	1 if M is in R or Q

3.2 OPTIMIZING ASSIGNMENTS MAIN MEMORY

In the following discussions, let E = number of word times required to execute a given command, and Se = the current sector at which the read heads are located. In all examples given, the operation code is represented as a 3-character mnemonic. An asterisk next to the operation code indicates that the flag bit is one. All FADAC locations are shown as one to three octal digits, followed by a period, followed by one to three octal digits to designate "channel-sector". (Leading zeros are not shown in any address.) Registers are coded as A, L, N, D.0, D.1, R.0 through R.17, and Q.0 through Q.17. Throughout the discussion all numbers are in octal, unless otherwise indicated. Recall that at any time, t, the read heads of all 64_{10} channels of main memory are simultaneously positioned at the same sector, Se, while the write heads are positioned two sectors before the read heads (Se − 2). In general, to optimize an operand in main memory, assign the operand to a sector to (Se + 1). To optimize an NI in main memory, assign the NI to any sector from (Se + 1) through (Se + E). If a given NI is less that (Se + E), then the next optimum operand (OP) must be $\geq$ (Se + E) + 1.

Example.

Time No.	Location	Op-Code	Operand	NI	Remarks
1	$Se_1 = 100.2$	CLA	100.3	102.3	E for CLA = 1 $\therefore$ Op = NI = $(Se_1 + 1)$
2	$Se_2 = 102.3$	ADD	100.4	102.4	E for ADD = 1 $\therefore$ Op = NI = $(Se_2 + 1)$
3	$Se_3 = 102.4$	MPY	100.5	100.10	E for MPY = 22 $(Se_3 + 1) \leq NI \leq (Se_3 + E)$
4	$Se_4 = 100.10$	ADD	100.27	120.27	$Op = [(Se_3 + E_3) + 1]$

STORE and LOAD Commands (M in Main Memory).

STA, STL, STN, STD.

These commands require that the contents of the named register be written into the operand, M. Since the write heads in main memory are at (Se − 2) at time t, when the instruction is read, they will be (Se − 1) at time t + 1 when execution is initiated. Therefore, M is optimum at Se − 1. The optimum NI is sector Se + 1. Note that given an STD command, the contents of D.0 are stored in M when M is even; the contents of D.1 are stored in M when M is odd.

STO and STP.

When M is in main memory, these commands require 3 word times to complete, with the actual storage (writing) into M occurring during the third word time. The write heads, which are located at Se − 2 at the beginning of execution of these commands, have turned through to Se + 1 at the time storage occurs. Therefore, M is optimum at Se + 1, and the NI is optimum at Se + 3 (or $Se + 1 \leq NI \leq Se + 3$) if the next operand is $\geq \left[(Se + 3) + 1\right]$.

STR.

This command is initiated by FADAC hardware by writing the contents of sector Se + i and Se + i + 1 of the R-loop into the D-loop, and then transferring the information from a sector of D into M. Because of the initial 2 word time delay during which the D-loop is filled, M is optimum at Se + 1. Again, the NI may be any sector between Se + 1 and Se + E, if the next operand is $\geq \left[(Se + E) + 1\right]$; where E = 22 word times.

Locations M through M + 17 are filled with the contents of the R-loop during the execution of this command.

LDR* and LDQ.

These commands transfer the contents of M through M + 17 sequentially into the corresponding sectors of the R or Q-loop. Therefore, M is optimum at Se + 1.

Example.

Line No.	Location	Op-Code	Operand	NI	Notes
1	Se_1 = 014.75	STA	76.74 (Se_1 - 1)	60.76 (Se_1 + 1)	(A) $\rightarrow$ 76.74
2	Se_2 = 060.76	STP	70.77 (Se_2 + 1)	62.101 (Se_2 + 3)	NI address in A $\rightarrow$ 70.77; new contents of 70.77 $\rightarrow$ N. E for STP = 3 word times NI = Se_2 + 3
3	Se_3 = 62.101	CLA	64.102	70.102	OP = NI = Se_3 + 1
4	Se_4 = 70.102	STO	130.103	70.105	OP address in A $\rightarrow$ 130.103* Modified M $\rightarrow$ N, E for STO = 3 word times, NI = Se_4 + 3
5	Se_5 = 70.105	STR	70.106	70.120	$Se_5 + 1 \leq NI \leq Se_5 + E$ E for STR = 22
6	Se_6 = 70.120	ADD	76.130	24.130	M = (Se_5 + E) + 1
7	Se_7 = 24.130	LDR*	100.131	24.150	M = Se_7 + 1; NI = Se_7 + E
8	Se_8 = 24.150	TRA	R.O	----	(See Section 3.6)

*Assume working storage size is 12. Channel 130 is HOT. (See Section 3.7.)

In all cases, optimization must be based on the current operand, whether it is optimized or not. For example, consider a case where an operand is optimized on first occurrence. Subsequent references to that operand usually cannot be optimized.

Example.

Line	LO	Op-Code	Op-Ad	NI
1	70.2	CLA	110.3	120.3
2	120.3	MPY	100.4	100.25
3	100.25	STA	110.24	70.26
4	70.26	ADD	110.3	100.3
5	100.3	STA	72.2	100.4

In line 4, with Se = 26, the program calls for the contents of 110.3, which were originally used and assigned an optimum location in line 1. The choice of optimum NI for line 4 is based on the location of the read heads at the time the operand is read, in this case, sector 3.

3.3 OPTIMIZING ASSIGNMENTS R & Q-LOOPS

Note that the main read head in the high speed loops, and the only write head are always located at Se, modulo 20, while the auxiliary read head is at (Se + 10). Thus at anytime, t, two sectors in R and Q, spaced 10 sectors apart, are simultaneously under a read head. Therefore, for all commands which do not require writing into a sector in R or Q, the programmer has a choice of two optimum sectors in the high speed loops. In contrast, for all STORE (write) commands only one R or Q-loop sector is optimum. In view of this, the following rules apply to optimizing assignments in the high speed loops:

1. In general, to optimize the R or Q operands of STORE instructions, assign the operand to sector (Se + 1), modulo 20.

2. To optimize operands for all other commands, assign the operand to one of the two sectors defined by (Se + 1), modulo 10.

3. To optimize an NI in R or Q, assign the NI to any sector between (Se + 1) and (Se + E), modulo 10. If a given NI is less than (Se + E), modulo 10, then the next operand must be $\geq$ Se + E + 1.

Example.

Line No.	Location	(Se) Read Head in Main Memory is @ Sector	Write and Read Heads In R (& Q) are @ Sectors:		Op-Code	Operand	NI
			Main Read & Write	Auxiliary Read			
1	110.30	30	10	0	CLA	R.11	R.1
2	R.1	31	11	1	ADD	R.12	R.2
3	R.2	32	12	2	MPY	R.13	R.4
4	R.4	54	14	4	ADD	102.55	R.5
5	R.5	55	15	5	STA	R.16	R.6
6	R.6	56	16	6	STN	Q.17	72.57
7	72.57	--	--	--	--	--	--

In line 1 above, the main R-loop read head picks up the operand in R.11, while simultaneously the auxiliary read head reads the contents of R.1. In line 2, precisely the same simultaneous reading of the operand and NI occurs. Because the MPY command of line 3 takes 22 word times to execute, the NI may be any sector between $Se_3 + 1 = 33$ and $Se_3 + E = (32 + 22)$, modulo 10, while the next operand (line 4) must be $Se_3 + E + 1 = (32 + 22 + 1) = 55$. Note that the operand of line 4 could have been optimally assigned to any sector 55 in main memory, or to sectors 5 or 15 in R or Q. In line 5, since the op-code is a STORE command, only sector 16 is the optimum operand in R or Q (i.e., 56, modulo 20).

The user is cautioned that when the operand address of a transfer or test command stored in R or Q is not located in R or Q, execution of the instruction will not be completed until the address of operand is searched and found. Therefore, for example, if one plans to affect the high speed execution of a loop which contains a test, the operand address of the test should be in the R or Q-loop, as follows:

Line No.	Control Location	First Time Thru Loop, Read Head in Main Memory is @ Sector	Write and Read Heads in R (& Q) are @ Sectors:		Op-Code	Operand	NI
			Main Read & Write	Auxiliary Read			
1	10.137	137	17	7	STA	70.136	R.0
2	R.0	140	0	10	CLA	R.1	R.11
3	R.11	141	1	11	SUB	R.2	R.12
4	R.12	142	2	12	STA	R.1	R.13
5	R.13	163	3	13	TZE	R.14	R.4
6	R.14	164	4	14	TRA	20.13	-

In the above, if R.13 were coded as "TZE 20.13, R.4", the test would not be executed until location 20.13 was found in main memory. Thus, each time through the loop as much as a whole drum revolution may be lost. In contrast, when coded as in the example, full advantage is taken of the high speed registers. This "costs" one extra instruction to transfer out of the high speed register. Note that the operand of the store instruction of line 4 caused an unavoidable loss of 20 word times since it was optimally assigned in line 2 above.

3.4 USE OF REGISTERS FOR TEMPORY STORAGE

Very few FADAC commands use all of the registers during execution. When the commands called in a program do not use a register, that register may be used as temporary storage for operands or instructions.

It is possible to store the contents of A into any register by using the proper command. Note that there is no direct "Store A into L" command. Instead, it is necessary to use a CLA* or CLS*, which cause the former contents of the A-register to be stored in the L-register, as illustrated in Line 5 of the following example:

Example.

Line No.	Op-Code	Operand	Notes
1	STA	D.1	(A) $\rightarrow$ D.1
2	STA	D.0	(A) $\rightarrow$ D.0
3	STA*	N	(A) $\rightarrow$ N
4	STA*	12.0	(A) $\rightarrow$ N and 12.0
5	CLA*	12.3	Former (A) $\rightarrow$ L; (12.3) $\rightarrow$ A
6	LDQ	A	(A) $\rightarrow$ All sectors of Q
7	LDR*	A	(A) $\rightarrow$ All sectors of R

It is possible to transfer the contents of D.0 or D.1 to A, L, N, R, Q, or main memory using the proper command or commands in the proper sequence, as illustrated below:

Line No.	Location	Op-Code	Operand	Next I	Notes
1	2.17	CLA	D.0	2.20	(D.0) $\rightarrow$ A
2	2.20	CLA*	A	2.21	(D.0) $\rightarrow$ L
3	2.21	STD*	M	2.22	(D.i) $\rightarrow$ M and N, where the D Sector stored depends on whether M is odd or even.
4	2.22	STD	R.3	2.23	(D.1) $\rightarrow$ R.3
5	2.23	STD	Q.4	2.24	(D.0) $\rightarrow$ Q.4
6	2.24	STD	10.23	2.25	(D.1) $\rightarrow$ 10.23
7	2.25	STD	60.24	2.26	(D.0) $\rightarrow$ 60.24
8	2.26	LDR*	D.0	2.46	D.0 $\rightarrow$ all even sectors, of R-loop D.1 $\rightarrow$ all odd sectors of R-loop

(1) The contents of D.0 or D.1 must be in the A-register in order to transfer them to the L-register. See Lines 1 and 2 of example above.

(2) D.1 and D.0 cannot be stored into each other. The instructions STD D.1 or STD D.0 will not store D.0 in D.1 or D.1 in D.0. The D-loop will remain unaffected.

(3) As illustrated in line 8, when a LDR* D.i (or LDQ D.i) command is executed, all even sectors of the R (or Q) loop will be filled with the contents of D.0, and all odd sectors will be filled with D.1.

As shown in the next example it is possible to transfer the contents of L or N to A, R, Q, D.0 or D.1, or main memory. L may be stored in N as in Line 5. Note that the STL instruction itself does not store L in the N-register. The flag is required.

Line No.	Location	Op-Code	Operand	Next I	Notes
1	4.0	STL	D.1	4.1	$(L) \rightarrow D.0$
2	4.1	STN	D.0	4.2	$(N) \rightarrow D.1$
3	4.2	STL	R.3	4.3	$(L) \rightarrow R.3$
4	4.3	STN	Q.4	4.4	$(N) \rightarrow Q.4$
5	4.4	STL*	14.3	4.5	$(L) \rightarrow N$ & 14.3
6	4.5	STL	2.4	4.6	$(L) \rightarrow 2.4$
7	4.6	STN	2.5	4.7	$(N) \rightarrow 2.5$
8	4.7	CLA*	L	4.10	$(L) \rightarrow A$; previous $(A) \rightarrow L$
9	4.10	CLA	N	4.11	$(N) \rightarrow A$
10	4.11	STL*	N	4.12	$(L) \rightarrow N$
11	4.12	STL	N	4.13	*

* No flag. No storage occurs, only a one word time delay is realized.

N cannot be stored directly into L. To accomplish this, do the following:

Line No.	Location	Op-Code	Operand	Next I	Note
1	2.1	CLA	N	2.2	(N) $\rightarrow$ A
2	2.2	CLA*	6.3	2.3	(6.3) $\rightarrow$ A, former contents of A = (N) $\rightarrow$ L

Note that the operand of Line 2 above may be any location or register.

Any of the registers may be used to store instructions or data for rapid access during execution. In writing routines using this feature, make certain that the desired contents of the registers used are not destroyed before being read. In the following assume location 40.2 contains some value, X; and location 40.1 contains the instruction MPY, N, Q.6.

Location	Op-Code	Operand	Next I	Notes
20.0	CLA	40.1	20.1	(MPY N. Q.6) $\rightarrow$ A
20.1	CLA*	40.2	44.2	(A) $\rightarrow$ L; X $\rightarrow$ A
44.2	ADD	A	R.3	(A) = 2X
R.3	STA*	D.1	L	(A) $\rightarrow$ D.1 and N
(L)	MPY	N	Q.6	(A) = $4X^2$
Q.6	ADD	D.1	20.27	(A) = $4X^2 + 2X$

Another method of using the registers for storage of instructions is shown in the following example:

Line No.	Location	Op-Code	Operand	Next I	Notes
1	2.0	CLA	2.1	2.2	$(2.1) \rightarrow A$
2	2.1	ADD	4.2	2.4	
3	2.2	CLA*	A	2.3	Former (A) $\rightarrow L$
4	2.3	TRA*	L	6.4	Transfer Instr to N
5	(L)	ADD	4.2	2.4	

Note that this example results in executing the instruction which was in line 2 from the L-register; and also causes the transfer instruction in line 4 to be stored in the N-register.

3.5 USE OF R AND Q-LOOPS

For rapid access, operands and instructions may be transferred from main memory to loops R and Q by using the instructions LDR* and LDQ. Information may be transferred back to main memory from R with the STR command.

Notice that there is no command to store the contents of Q in main memory. This can be accomplished by:

Location	Op-Code	Operand	Next I
2.17	LDR*	Q.0	2.37
2.37	STR	2.42	10.62

In the above, the contents of Q are first loaded into R; R is then loaded into main memory.

R and Q are loaded from one channel only. For instance, the instruction LDR* 2.177, go to 4.10, begins loading R from location 2.177. The next location loaded into R is 2.0 not 4.0. All information destined for R or Q is taken from 16 consecutive words of the same channel.

The same restriction is true of the STR command. It causes storage into one channel. The instruction STR 2.177 begins storing at 2.177. The next word stored from R is into 2.0. The loops will always be stored into 16 consecutive words of the same channel of main memory.

It is possible to store R into R with an instruction such as:

STR R.16

The contents of R will be displaced by two words; that is, R.16 will appear in R.0, R.2 will appear in R.4, etc. The last two transferred words will also appear in D.0 and D.1.

R and Q can be loaded from any of the registers. The result is that the loop will be loaded with the same number in all of its cells, if loaded from A, L or N; or with the two numbers in D.0 and D.1, if loaded from the D-loop.

Location	Op-Code	Operand	Next I	Notes
6.2	CLA	4.3	6.4	(A) = X
6.4	LDR*	A	6.24	(R.0 – R.17 = X)

3.6 INSTRUCTION MODIFICATION

For greater flexibility in a program it is often necessary to modify certain instructions. In FADAC, this can be done using standard techniques involving standard commands. The process can be greatly simplified, however, by using the commands STORE OPERAND ADDRESS (STO) and STORE PROGRAM ADDRESS (STP), which replace the contents of one portion of an instruction in memory without affecting any other portion.

Care must be taken to introduce a delay between the time a store command which modified an instruction, is called, and the time the modified instruction itself is called. Both STO and STP require 3 word-times for execution, if M is in main memory, or 1 word time if M is in R or Q. Thus, for example, the following will cause execution of the instruction in Location 72.5 before the Operand Address in that location is modified, since the contents of the NI address are sent to the X-register at the same time that execution of the STO command is initiated. (See Figure 3.1 - Multiplexing Diagram.)

Example:

Location	Op-Code	Operand	NI
2.4	STO	72.5	72.5

To properly modify an instruction, and immediately follow the modification with execution of the modified instruction, an extraneous instruction must be coded, such as:

Line No.	Location	Op-Code	Operand	NI
1	2.4	STO	72.5	2.7
2	2.7	TRA	N	N

As noted in Chapter 2, given an STO or STP command, the modified contents of M (in this example M = 72.5) are stored in N. The TRA N of line 2 then causes execution of the modified instruction from the N-register.

Line 2 may be coded:

Line No.	Location	Op-Code	Operand	NI
2 (a)	2.7	CLA	A	72.5
		or		
2 (b)	2.7	TRA	72.5	

Both 2(a) and 2(b) will cause execution of the modified instruction in 72.5.

The same precaution must be taken in the use of all load or store commands, as illustrated in the following examples:

Line No.	Location	Op-Code	Op-Address	NI
1	110.1	CLA	110.2	112.2
2	112.2	ADD	110.3	112.3
3	112.3	STA	110.4	110.4
4	Etc. . . .			

Execution of the command in line 3 will cause the unmodified contents of 110.4 to be sent to the X-register. To avoid this, the program should be corrected as follows:

Line No.	Location	Op-Code	Op-Address	NI
3	112.3	STA	110.4	112.4
4	112.4	CLA	A	110.4

Line 4 is a do-nothing command inserted to allow modification of the contents 110.4 prior to transfer of control to that location. Consider the effect of coding a LDR* or LDQ command as follows:

Line No.	Location	Op-Code	Operand	NI
1	70.1	LDR*	70.2	R.0
2	100.27	LDQ	30.130	Q.7

In line 1, the contents of locations 70.2 thru 70.21 will be loaded into the R-loop. However, at the initiation of execution of this command the present unmodified contents of R.0 (or any R sector designated) are stored in the X-register, and it is this unmodified command which is executed next. Precisely the same sequence of events occurs when the command in line 2 is executed. This may be desired. However, to execute the modified contents of a loop sector, time must be allowed for completion of the LDR* or LDQ command before control is transferred to the desired loop sector, as:

```
Line 1  70.1 LDR* 70.2, 72.21
Line 1a 72.21 CLA  A, R.0
Line 2  100.127 LDQ 30.130, 100.47
Line 2a 100.47   TRA  Q.7
```

In lines 1 and 2, the NI of the load instruction is coded $\geq$ 17 word times past the operand sector, to introduce the necessary delay. Transfer of control to the desired loop sector is achieved in line 1a by a "do-nothing" CLA A with a loop sector NI; and in line 2a with an unconditional transfer to a loop sector.

3.7 STORE INSTRUCTION RESTRICTIONS

Because the Write heads of all channels designated as permanent storage (COLD) are de-energized when the MLU is not connected to FADAC, the programmer must exercise care in the use of all STORE instructions, such that either (1) the Operand of a STORE instruction is in HOT memory or, (2) the information is temporarily stored in a register, then either retrieved from the register and sent to HOT memory for later use, or executed from the register before the contents of that register are changed.

In the following example, assume working storage size is 4 channels (i.e., channels 70–76 only are HOT).

Example.

Line No.	Location	Op-Code	Operand	NI
1	2.60	CLA	4.61	2.61
2	2.61	ADD	4.62	2.62
3	2.62	STA	74.61	2.63
4	2.63	STO	4.64	2.66
5	2.66	STN	74.67	R.0

In line 3, the operand is in HOT memory, so that proper storage occurs. In line 4 the instruction in 4.64, modified by the operand in the A-register is sent to N, and in line 5, N is sent to working storage for later use. The contents of 4.64 are not modified.

See Section 2.2.2 for a detailed description of all store instructions.

3.8 LINKAGE TO SUBROUTINES

Entry to a subroutine and subsequent return to the main program is facilitated by using the transfer instructions in conjunction with the STORE PROGRAM

ADDRESS command. In the following example, the entry to the subroutine is made with a transfer instruction in the main program. The NI of this instruction is the return address back to the main program. Upon execution of the transfer instruction, the flag bit puts the entire transfer instruction into N, and control goes to the subroutine. The first instruction of the subroutine will take the return address (NI) out of N and put it into the last instruction of the subroutine; and simultaneously store the argument in A into the L-register. Upon completion of the subroutine, control returns to the main program.

Example.

Line No.	Location	Op-Code	Operand	NI	Notes
			Main Program		
1	64	CLA	(Argument)	64.1	Argument A
2	64.1	TRA*	6.10	64.22	(Transfer Instruction N)
3	64.22	.	.	.	Re-entry to Main Program (MP)
	.	.	.	.	
	.	.	.	.	
			Subroutine		
4	6.10	CLA*	N	6.11	Argument L Transfer Instruction A
5	6.11	STP	072.21	6.12	Return to MP stored in last instruction of subroutine
6	6.12	CLA	L	.	Argument A
	.	.	.	.	
	.	.	.	.	
	.	.	.	.	
7	72.21	STD	074.22	(64.22)	Re-enter MP thru NI of last instruction of subroutine as modified by the instruction in line 5

It is sometimes desirable to store a complete word into the last location of the subroutine, rather than the return address only:

Line No.	Location	Op-Code	Operand	Next I	Notes
1	64.1	TRA*	6.10	64.22	(N) = TRA* 6.10, 64.22
2	64.22	CLA	--	--	Re-entry to MP
			Subroutine		
3	6.10	CLA	N	6.11	(N) A
4	6.11	STP	6.12	4.15	Contents of 6.12, modified by the NI now in A are sent to N
5	6.12	STD	74.15	(0000)	
6	4.15	STN	72.21	6.15	Contents of N stored in last location of Subroutine, with its NI set for return to MP
7	6.15	MPY	.	.	
			.	.	
			.	.	
8	72.21	STD	74.16	64.22	Return to MP

In this example, assume that the MLU is disconnected at execution time. Therefore, Channel 6 is in cold memory, and cannot be written into. Line 1 causes transfer to a subroutine which begins in Channel 6. The first word of the subroutine (line 3) brings the transfer instruction to A. The STP command of line 4 sends the contents of 6.12 to the N-register, with the NI in N changed to contain the NI currently in the A-register. Thus, on execution of the command in line 4, the N-register contains

Op-Code	Operand	NI
STD	74.16	64.22

where the NI address (64.22) is the return to main program. The contents of 6.12 remain unchanged. The command in line 6 stores the modified contents of N into the last word of the subroutine (line 8).

3.9 TABLE LOOK-UP

The FADAC instruction repertoire includes two table look-up commands, Equal Search (EQS) and Greater

Than or Equal Search (GES), both of which operate in a similar manner. (See Section 2.2.5.) To illustrate a typical application, consider the following equation:

$$Y = B_xX + X^2$$

Assume the value of the coefficient Bx to be used in evaluating this equation depends on the computed value of X. If a prestored value of X is greater than or equal to the computer value of X, then a particular value of the coefficient Bx is to be submitted in the equation. The programmer must construct a table for the variable coefficients such that each stored value of X is followed by the value to be assigned to the coefficient. Consider the structure of a table for the value of B_x. Values of X are stored in the table in ascending order, each followed by the B_x to be used with that computed X which is within a given range of values. X is scaled at 0; B_x at (–2).

Location	FADAC Octal Word	Remarks
2.170	37757000000	$X = -0.002$
2.171	00015770000	$B_x = 0.000427$
2.172	37767600000	$X = -0.001$
2.173	00013150000	$B_x = 0.000342$
2.174	00010140000	$X = 0.001$
2.175	00004061000	$B_x = -0.000125$
2.176	00071260000	$X = 0.007$
2.177	00001424400	$B_x = 0.000047$

Assume that in our problem, X has been evaluated as 0.004. The following program segment will search for the proper value of B_X:

Location	Op-Code	Operand	Next I	Notes
4.20	CLA	MASK	4.21	(A)= 37777777777= MASK
4.21	CLA*	X	4.22	(A) = X = 0.004 (L) = 37777777777
4.22	GES	2.170	4.00	Search table for $X \geq 0.004$
4.00	STA	76.1	4.01	(A) = B_x = 0.000047 Store B_x in 76.1

The GES command searches the table until it finds a word that is greater than or equal to X, at which time the next word in the table is read into A. When the proper B_X has been found, control is transferred to the next instruction. Note that L is filled with the mask before the Search is initiated.

The last sector searched in any channel is 176. Therefore, if the proper X is not found before the search reaches the end of the channel, the A-register will contain the same number as it did at the beginning of the search.

3.10 MEMORY II AND MEMORY III ADDRESSING TECHNIQUE

In its original design, FADAC memory contained only 32 channels plus the A, N, L, D, R, and Q-registers. When the design was expanded to the present 64 channel version, it became necessary to devise a special method to address all channels and registers without increasing the size of the FADAC word. Since a 32 bit FADAC word allows only 7 bits (see Section 2.1) to identify a channel address, an operand or NI address in channels numbered 100 through 136 (Memory II) has the same bit configuration as addresses in channels 300 through 336 (Memory III). To enable the computer to identify the desired channel in these ambiguous areas, FADAC hardware includes a flip-flop which is set to its OFF state to interpret channel address 100 - 136 as Memory II and Reset to its ON state to interpret the same bit configuration as Memory III.

A 146 in the channel address of the NI of a given instruction is required to change the state of this flip-flop. Initially, when FADAC is turned on or when any control button on the front panel of FADAC is depressed, the flip-flop is set for Memory II addressing. The first 146 NI channel address sensed will set the flip-flop to its ON state, either permanently when the 146 NI is used with a TRA operation code, or momentarily when the 146 NI is coded with any other operation code. In all cases, when a 146 NI channel address is coded with a TRA operation code, the change in the state of the memory control flip-flop remains fixed until the next TRA-146 combination is sensed, or until a control button is depressed. In contrast, a 146 NI channel address coded with a non-TRA operation code causes a change in the state of the flip-flop which lasts only until the instruction is executed. See Section 3.11.

No 146 NI is required to call or transfer control to the registers A, N, L, D, R, or Q.

Within a FADAC word, any reference to a channel in Memory III is coded as 100 through 136. The leading digit, 3, of a channel address is only used when the location field is specified, as on an input tape or on keyboard entry. In these cases the five octal digit address must be followed by the octal location code (a period). (See Section 2.2.7,)

Case I through Case IV specify the rules which must be used to properly address Memory II and III.

Case I - Location Field in Memory I.

Memory Control flip-flop set to its OFF state to interpret channels 100 through 136 as Memory II.

a. To call an operand in Memory I or II, no 146 is required.

Example. 1a.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
72.4	CLA	100.5	74.5	07204.	07405250005
74.5	ADD*	70.5	72.6	07405	27206007006
72.6	STA	70.5	72.7	07206.	07207507005
72.7	STA	100.5	70.10	07207.	07010510006

b. To call an operand in Memory III, a 146 must appear as the NI channel address, to momentarily set the flip-flop to its ON state.

Example 1b.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
00.21	CLA	300.22	146.22	00021.	14622250022
00.22	ADD.	300.23	146.23	00022.	14623010023
00.23	ADD	70.24	72.24	00023.	07224007024
72.24	STA	70.25	000.25	07224.	00025507025

c. To transfer control to Memory I or II, no 146 is required with a TRA operation code.

Example 1c.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
00.136	CLA	70.137	72.137	00136.	07337247137
72.137	TRA*	100.140	00.026	07337.	20026150140
100.140	.	.	.	10140.	.
	.	.	.	.	.
	.	.	.	.	.
40.06	TRA	74.07	00.00	04006.	00000147407

d. To transfer control to Memory III, the flip-flop must be reset to its ON state. Therefore, a 146 must appear in the NI field with a TRA operation code.

Example 1d.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
20.25	CLA*	70.26	20.26	02025.	$\underline{2}$2026247026 (Note 1)
20.26	TRA*	300.27	146.27	02026.	$\underline{3}$4627150027 (Note 1)
300.27	ADD	300.30	320.30	30027.	12030010030

Note 1:

Recall that the first digit of a FADAC octal word is the sum of the flag bit and the first channel address bit.

Case II - Location Field in Memory I.

Memory Control flip-flop set to its ON state to interpret channels 100 through 136 as Memory III (channels 300 through 336).

a. To call an operand in Memory I or III, no 146 is required.

Example IIa.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
6.0	CLA	300.1	72.1	00600.	07201250001
72.1	ADD	4.2	300.2	07201.	10002000402
300.2	ADD	300.3	40.3	30002.	04003010003

b. To call an operand in Memory II, a 146 must appear as the NI channel address to momentarily reset the flip-flop to its OFF state.

Example IIb.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
70.5	SUB*	100.6	146.6	07005.	34606030006
70.6	ADD	102.7	146.7	07006.	14607010207
70.7	MPY	110.10	146.32	07007.	14632211010
70.32	---	---	---	07032.	-----

c. To transfer control to Memory I or III, no 146 is required.

Example IIc.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
6.107	CLA	310.110	70.110	00707.	07110251110
70.110	TRA*	306.111	70.116	07110.	27116150711
306.111	TRA	04.121	0.0	30711.	00000140521

d. To transfer control to Memory II, a 146 must appear as the NI channel address with a TRA operation code.

Example IId.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
22.6	CLA	320.7	24.7	02206.	02407252007
24.7	TRA*	100.10	146.10	02407.	34610150010
100.10	CLA	136.11	100.11	10010.	10011253611

Case III - Location Field in Memory II.

Memory Control flip-flop set to its OFF state to interpret channels 100 through 136 as Memory II.

a. To call an operand in Memory I or II, no 146 is required.

Example IIIa.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
102.57	TOV	50.60	106.60	10257.	10660165060
106.60	ADD	126.61	130.61	10660.	13061012661
130.61	STA	70.60	100.62	13061.	10062507060

b. To call an operand in Memory III, a 146 must appear in the NI channel address, since the flip-flop must be momentarily reset to its ON state.

Example IIIb.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
134.7	CLA	310.10	146.10	13407.	14610251010
134.10	ADD	70.11	130.11	13410.	13011007011
130.11	ADD	334.12	146.12	13011.	14612013412
130.12	MPY	306.13	146.34	13012.	14634210613
130.34				13034.	

c. To transfer control to Memory III, a 146 must appear in the NI channel address with a TRA operation code.

Example IIIc.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
126.130	CLA	100.131	124.131	12730.	12531250131
124.131	TRA	302.132	146.00	12531.	14600150332
302.132	ADD	306.133	302.133	30332.	10333010733
302.133	---	---	---	30333.	----

d. To transfer control to Memory I or II, no 146 is required with a TRA operation code.

Example IIId.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
126.0	CLA	110.1	100.1	12600.	10001251001
100.1	TRA	70.2	0.0	10001.	00000147002
70.2	TRA*	136.3	0.0	07002.	20000153603
136.3	TRA	106.4	0.0	13603.	00000150604

Case IV - Location Field in Memory III.

Memory Control flip-flop is set to its ON state to interpret channels 100 through 136 as Memory III.

a. To call an operand in Memory I or III, no 146 is required.

Example IVa.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
306.21	CLA	76.22	304.22	30621.	10422247622
304.22	ADD	00.23	306.23	30422.	10623000023
306.23	STA	70.22	306.24	30623.	10624507022

b. To call an operand in Memory II, a 146 must be coded in the NI field.

Example IVb.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
324.02	MPY	100.03	146.24	32402.	14624210003
324.24	ADD*	110.25	146.25	32424.	34625011025
324.25	STA	70.24	306.26	32425.	10626507024

c. To transfer control to Memory I, or III, no 146 is required with the TRA operation code.

Example IVc.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
336.130	CLA	300.131	322.131	33730.	12331250131
322.131	TRA	306.132	0.0	32331.	00000150732
306.132	TRA*	10.133	0.0	30732.	20000141133
10.133	TRA	314.134	0.0	01133.	00000151534

d. To transfer control to Memory II, a 146 must appear in the NI field with a TRA operation code.

Example IVd.

				FADAC Octal	
Lo	Op-Code	Operand	NI	Lo	Word
320.10	CLA	R.11	320.11	32010.	12011254211
320.11	TRA	100.12	146.12	32011.	14612150012
100.12	ADD	120.13	122.13	10012.	12213012013
122.13	ADD	R.14	D.1	12213.	16001014214
(D.1)	TRA	322.14	146.00	(executed	from D.1)
322.14	STA	D.0.	70.22	32214.	07022516000
70.22	TRA	Q.3	L	07022.	17200155203

Example IVd also illustrates that no special coding is required to address the registers A, L, N, R, Q, or D from Memory I, II or III, regardless of the state of the memory control flip-flop.

Note that control may be in Memory I regardless of the memory control flip-flop (see Cases I and II); while control can only be in Memory II when the flip-flop is set to its OFF state, and in Memory III when the flip-flop is set to its ON state.

Channel Return Given a 146 - NI Channel Address.

As each instruction is processed, its NI is automatically stored in a special NI address buffer. The contents of the buffer are updated to a new NI address when execution of an instruction is completed and the next instruction read. However, when the new NI contains a 146 channel address, the contents of the channel portion of the NI buffer remain unchanged; that is the last non-146 NI channel address remains in the NI address buffer until a new non-146 is sensed. During execution of each instruction, FADAC hardware examines the NI buffer, and causes transfer of control to the channel sector specified when execution is completed.

Example.

Line No.	Location Field	Op-Code	Op-Address	NI	Notes
1	0.0	CLA	110.1	100.1	Program is started by Depressing SM (LOCATION 0.0) FF is set OFF (All 100 through 136 references are interpreted as Memory II.)
2	100.1	ADD	70.2	70.3	
3	70.3	STA	70.4	72.4	
4	72.4	TRA	110.2	000.0	Transfer from Memory I to Memory II - No 146 necessary since FF is set to interpret 100 through 136 as Memory II by line 1.
5	110.2	ADD	130.3	72.3	
6	72.3	ADD	330.4	146.4	Momentarily reset FF to ON to pick up operand in Memory III.
7	72.4	TRA	330.5	146.0	FF reset to ON to transfer control to Memory III.
8	330.5	MPY	300.6	70.27	No 146 necessary to return to Memory I.
9	070.27	STA	70.26	300.30	No 146 necessary to return to Memory III, since FF set ON by line 7.
10	300.30	ADD	100.31	146.31	Momentarily set FF to OFF to get Operands from Memory II. Return to channel 300 as each instruction is executed. executed.
11	300 31	ADD	104.32	146.32	
12	300.32	ADD	110.33	146.33	
13	300.33	ADD	120.34	146.34	
14	300.34	TRA	110.36	146.00	FF reset to OFF to transfer control to Memory II.
15	110.36	STA	70.36	70.37	
16	70.37	TRA	110.50	100.50	No 146 necessary since FF set OFF by line 14.

In the example above, computation is initiated by depressing the Sample Matrix button (location 0.0), and the NI address buffer is filled with 100.01. When the instruction in line 1 is executed, control is sent to 100.01. As each instruction in lines 2 through 5 is executed, the contents of the NI buffer are updated. However, when the instruction in line 6 is read, only the sector portion of the NI buffer is changed. Thus, during execution of the line 6 instruction, the NI buffer contains 72.04. When execution is completed, control is transferred to location 72.04 and the contents of the NI buffer changed to 72.00. Since line 7 contains a TRA instruction, the contents of the NI buffer are ignored, and control is transferred to the operand -- 330.05. After execution of the instruction in line 8, 300.30 is stored in the NI address buffer. The channel portion of the buffer remains unchanged throughout the execution of the instructions in lines 10 through 14, since each contains a 146 NI channel address. Note that the return to the last non-146 NI channel is automatic, but that the desired sector must be coded.

3.11 USE OF 146 NI

When a TRA-146 combination is sensed, transferring control to the opposite side of memory, and information is immediately desired from the original side of memory, special care must be taken to keep track of the last non-146 NI sensed. Consider the following example.

Example.

Lo	Op-Code	Op-Ad	NI-Ad	Line #	Remarks
120.177	STA	70.176	110.2	1	
110.0	TRA	300.1	146.0	2	Reset flip-flop to Memory III
300.1	CLA	100.2	146.2	3	
310.2	ADD	---	---	4	

Note that the location field of line 4 is in channel 3̲10. This is true since the last non-146 NI channel address, 110, appearing in line 1, is interpreted as 3̲10, because of the TRA-146 coded in line 2. In contrast, consider:

Lo	Op-Code	Op-Ad	NI-Ad	Line #	Remarks
120.177	STA	70.176	72.0	1	
72.0	STRA	300.1	146.2	2	Reset flip-flop to Memory III
300.1	CLA	100.2	146.2	3	
72.2	ADD	300.3	302.3	4	
302.3	----	---	----	5	

Here the location field of line 4 is in channel 72, which is the last non-146 NI channel address sensed. Since no ambiguity exists in Memory I addresses, the channel address in the NI buffer, 72, is not affected by the TRA-146 in line 2.

If the programmer intends a return to channel 300, in the above examples, then a dummy instruction must be inserted to reset the NI buffer before a call for information from the original side of memory:

Lo	Op-Code	Op-Ad	NI-Ad	Line #	Remarks
120.177	STA	70.176	110.0	1	
110.0	TRA	300.1	146.0	2	Reset flip-flop
300.1	STA	A	<u>300.2</u>	3	Dummy instruction to Reset NI buffer
300.2	CLA	100.3	146.3	4	Clear and add contents of 100.3 and go to 300.3
300.3	ADD	302.4	300.4	5	Lo field interpreted as 303.3 as desired.

Whenever a Test for Zero (TZE) or Test for Plus (TPL) (branching) instruction is used, special precautions must be taken:

Example.

Location of Command	Op-Code	Op-Address	Next Instr' Address	Remarks
300.15	CLA	320.16	300.17	
300.17	MPY	70.20	300.20	
300.20	ADD	320.42	300.43	
300.43	TPL	310.44	312.44	If plus go to 310.44 If minus go to 312.44

Note that the command in location 300.43 will cause transfer to channel 310 if the result of the computation is plus, or to channel 312 if the result is minus. In either case, 312.44 is sensed as the Next Instruction address, and is therefore stored in the NI channel buffer. Observe the confusion if the branching instruction is followed by a call for information stored on the opposite side of memory, as shown in the following:

Example.

Location of Command	Op-Code	Op-Address	Next Instr' Address	
300.43	TPL	310.44	312.44	
312.44	CLA	100.45	146.46	
312.46	STA	70.45	312.47	
312.47	HLT	---	0.0	
.	.		.	
.	.		.	
.	.		.	
310.44	CLA	102.45	146.46	Control goes to 312.46
310-46	ADD	300.47	310.47	
310.47	STA	300.48	310.48	
			etc.	

Should the computation yield a negative result, no problems exist, since the information in 100.45 will be sent to the A-register, after which control will return to channel 312, sector 46 as desired. However, if the result is positive, a transfer to 310.44 will occur, the contents of 102.45 will be sent to the A-register and since the NI channel buffer still contains the 312 sensed as the last non-146 NI address, control will then go to 312.46, rather than to the 310.46 desired, yielding a program error. Since the programmer cannot usually predict which branch will be taken, he is required to either (1) program the same channel for each branch of a conditional transfer, or (2) avoid a 146 NI channel reference immediately following a conditional transfer instruction by the insertion of a dummy instruction which will change the address stored in the NI channel buffer, as follows:

Example.

Location of Command	Op-Code	Op-Address	Next Instr' Address	Remarks
300.43	TPL	310.44	312.44	
312.44	CLA	100.45	146.46	Contents of 100.45 to A-reg. Go to 312.46.
312.46	STA	70.45	312.47	Store A in 70.45 Go to 312.47
.	.	.	.	etc.
.	.	.	.	
.	.	.	.	
310.44	CLA	A	310.45	Dummy instruction changes the NI channel buffer to 310.
310.45	CLA	102.45	146.46	Clear & add contents of 102.45. Go to 310.46.

(Continued)

Example. (Continued)

Location of Command	Op-Code	Op-Address	Next Instr' Address	Remarks
310.46	ADD	300.47	310.47	Add contents
.	.	.	.	of 300.47 to
.	.	.	.	A-Reg. Go to
.	.	.	.	310.47 etc.

3.12 WORD INPUT/OUTPUT

The rules specified for designation of Memory II and Memory III are applicable in the execution of all commands except Parallel I/O by word. Recall that these commands require that the L-register be programmed to contain the starting location into which input is to be stored, or from which output is to be fetched. When input is in Octal code, use of the 146 to differentiate between Memory II and Memory III locations is ineffective. Rather, the value of bit 18 in the L-register when an Enter code is sensed controls FADAC's interpretation of the storage location. When this bit equals 0, the starting location is interpreted as in Memory I or II. When bit 18 of L equals 1, the storage location is interpreted as in Memory III.

For all other Parallel I/O, it is recommended that the following procedure be adopted:

1. Program all Parallel I/O Commands into Memory I.

2. Code the NI of all Parallel I/O Commands into Memory I.

3. Program a zero to L before setting up the starting location in L.

4. If the memory control flip-flop is set to interpret all 100 channels as in Memory II and the starting store (for READ) or fetch (for WRITE) location is in Memory III, code the following sequence of instructions.

a. TRA (to a Memory I lo), 146 XX ·

b. Set L to the starting Memory III location.

c. Code the desired Read or Write Command with an NI in Memory I.

Example.

Control is in Memory II. Read External Device, 9 words, in BCD code, storing the input into location 306.10 through 306.2 and return to 104.27.

```
  Lo     Op-Code     Operand        NI          Explanation
102.21    ZEL           0      ,  102.22          0      L
102.22    TRA         72.23    ,  146.0   Set FF for Memory III
 72.23    CLA         40.24    ,   72.24
 40.24    OCT     (00000030610)           Starting Lo Address
 72.24    CLA*          A      ,          L =00000030610
 72.25    RED*         10      ,   72.26  Read 9 words
 72.26    TRA        104.27    ,  146.0   Return control to
                                          Memory II
104.27     .
           .
           .
```

If the memory control flip-flop is set to interpret all 100 channels as in Memory III, and the starting store or fetch location is in Memory II, precisely the same sequence of instructions should be coded.

Example.

Control is in Memory III. Write to an External Device, 18 words in Alpha 5 code, starting at location 105.44, and return to 310.27.

```
  Lo     Op-Code     Operand        NI          Explanation
304.21   ZEL          0       ,  304.22          0      L
304.22   TRA          42.23   ,  146.0     Set FF for Memory II
 42.23   CLA          50.24       54.24
 50.24   OCT       (00000010544)            Starting Lo Addr.
 54.24   CLA*         A       ,   54.25    L = 00000010544
 54.25   WE5*         21      ,   54.26    Write 18 words
 54.26   TRA         310.27   ,  146.0     Return control to
                                           Memory III
```

If this procedure is used, the danger of loss of control of the setting of the Memory control flip-flop during I/O is eliminated.

CHAPTER 4

FADAC OPERATING PROCEDURES

4.1 CONTROL PANEL

Functional Description.

The computer control panel is shown pictorially in Figure 4.1. Primarily, the control panel is divided into five basic functional areas as follows:

1. Controls and Indicators
2. Keyboard Input Device
3. Mechanical Tape Reader
4. Matrix Input Device
5. Nixie Display Panel

The following is a summary of the functions of the operating controls and indicators of FADAC.

PWR ON, PWR OFF Switch.

When flicked upward, this switch actuates the power control circuits in the computer, energizing in the proper sequence the power supplies, blowers, and memory. When pulled down, the computer is de-energized.

PWR READY (Indicator).

This neon indicator lights after a delay (approximately 20 seconds) suitable for insuring the memory is fully operational. The light remains in its ON condition until the computer power is turned off. When the computer is placed in a Marginal Test Condition, the indicator blinks until normal operation mode is restored.

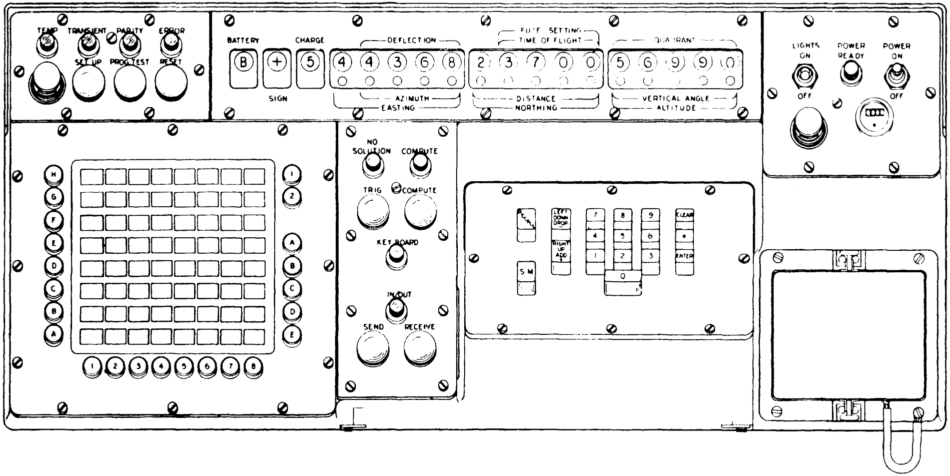


Figure 4.1. Production Engineering FADAC

Transient (Indicator).

Normally ON, this neon indicator blinks when a line voltage transient has occurred, or when the input voltage exceeds high or low tolerances. If the blinking cannot be reset with the RESET button, the input voltage is approaching either a low or high voltage kickout condition.

Temp. (Indicator).

Normally ON, this neon indicator blinks when the internal temperature of the machine is approaching marginal operation. The computer may usually be used, however, until temperature kick out actually occurs. Any system application of the computer must insure that the control panel and computer filters are accessable for cleaning, and are provided with unimpeded air flow. Additionally, a cut-out switch is provided on the rear of the computer to de-energize one of the internal blowers for cold temperature operation. This feature is also used as a maintenance tool to determine if proper blower action is present when a temperature warning occurs.

Parity (Indicator).

Normally ON, this neon indicator blinks when a parity error is detected, either when reading from memory or in reading from an input device in Fieldata or other parity-sensitive code. To clear the blinking, press the RESET button. If this does not correct the condition, turn FADAC OFF, then ON again, and proceed as desired.

Error (Indicator).

Normally ON, this neon indicator blinks when an overflow error has occurred in computation or when an error is sensed with the computer in the VERIFY mode.

No Solution (Indicator).

Normally ON, this indicator blinks when a No Solution Light (NSL) Command has been executed. It will continue to blink until one of the Discreet I/O Commands (DOF, OD1, OD2, OD3, ID1, ID2, or ID3) is sensed. The indicator may be programmed to blink to indicate an insoluble problem, or that an incorrect procedure has been employed.

Compute (Indicator).

Normally OFF, this neon indicator lights during the periods when the computer is in a computing state. It may also be turned ON and OFF under program control with execution of an Initiate Compute Mode (ICM) or Halt Compute Mode (HCM) instruction. Thus, the the light can be kept off while the computer is in a compute mode. This function is usually reserved for periods when the computer is either self-testing the memory arithmetic loop registers, or scanning the discrete input/output lines for evidence of an input from (or output to) an external device.

In/out (Indicator).

Normally OFF, this neon indicator lights when information is being transferred to or from an external device. It is automatically controlled by the execution of any computer READ or Write Command.

Keyboard (Indicator).

Normally OFF, this neon indicator lights when the computer calls for information from the keyboard. The light is automatically controlled by the execution of a READ KEYBOARD command.

Time Meter.

A running time meter is provided on the control panel to show total elapsed operation time of the computer.

Marginal Test (Switch).

While not a physical part of the control panel, a five position switch is provided on the left side of the computer main frame. Normally OFF, this switch provides four different combinations of marginal voltage. When the switch is placed in any of the marginal test positions, the PWR READY indicator will flicker.

Reset (Switch).

This switch, when momentarily depressed, resets the computer to the program or manual Halt mode from the Compute or Input/Output modes. It is used when a parity or overflow error has occurred; or to reset the Temperature Warning or Transient neon indicators when these are blinking.

Input Matrix.

The matrix consist of 64 windows arranged in an 8 x 8 array. The buttons used to select a specific input windows, corresponding to an assigned input parameter, are located on the left-hand column (lettered A through H) and on the bottom row (numbered 1 through 8). Selection of a given window sets binary information into bit positions 25 through 30 of the A-register, when a DIA command is executed (see Section 2.2.6). In addition to the 8 x 8 matrix there are several auxiliary buttons on the Matrix panel as described below:

a. Five buttons on the right hand bank (lettered A through E) which set binary information into bit positions 19 through 23 of the A-register when a DIA command is executed. These buttons are wired directly to the Battery tube of the Nixie display.

b. Two switches on the right hand bank (numbered 1 and 2) which set binary information into bit position 24 of the A-register when a DIA command is executed.

Mechanical Tape Reader.

The mechanical tape reader is used for entering data into FADAC. The input code may be either a 5 level teletype code or an 8 level Fieldata code. Starting or stopping of the tape reader may be internal under program control, or external by depressing the proper sequence of buttons. The running speed of the tape reader is approximately 10 characters per second.

Keyboard.

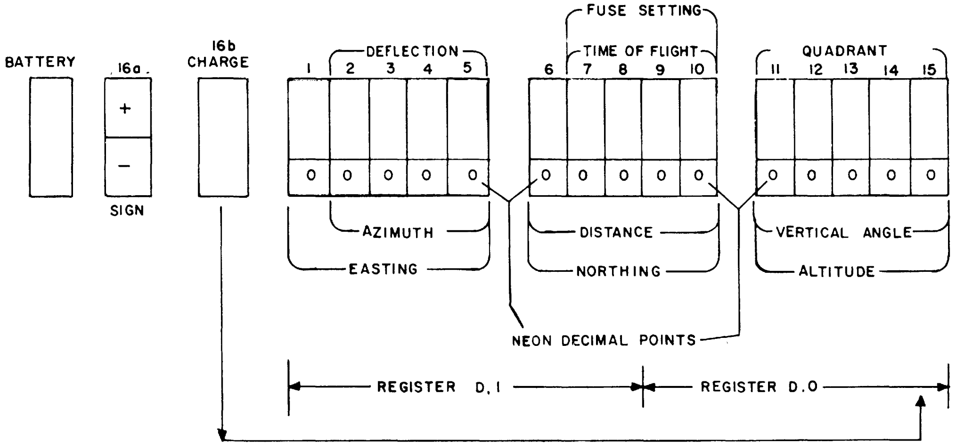
The computer keyboard is used for manual input of information. It contains keys for the digits 0 through 9, one key each for + and –, and three control keys (Clear, ., and Enter).

Control Buttons.

If the computer has been set into the program Halt state via the program, or by activation of the RESET switch, the control buttons may be used to reinitate computation. The buttons serve primarily as a hardware "bootstrap" to initiate execution of a program.

Nixie Display.

The nixie display is a programmed visual display. Information to be displayed must first be converted into Binary Code Decimal characters and stored properly in the 2 word D-loop. The display consists of 17 nixie tubes and 16 neon lamps used for decimal point display. Figure 4.2 is a diagram showing the relationship between the Nixie Lights and the D-register. (The Nixie tubes are numbered in the illustration to facilitate discussion). As shown, Nixie tubes 1 through 9 display the contents of the 32 bit positions in D.1; tubes 9 through 15 display the 28 most significant bit positions in D.0. Tubes 16a and 16b are both connected to the four least significant bit positions of D.0 (bits 28-31).



NOTE: The 4 least significant bits in D. O will energize either the sign or the charge nixie, but not both.

Figure 4.2. Relationship Between Nixie Lights and D-Register

If these bit positions contain a digit (BCD 0 to 9), tube 16b (the CHARGE nixie) will be activated. If they contain a sign (BCD 12 or 13), tube 16a (the SIGN nixie) will activate. A sign coded in any position other than the four least significant bits of D.O will be interpreted as a blank. Therefore, care must be taken to properly position a sign which is to be displayed. (See Appendix C for a list of BCD codes.) The Nixie tube labelled BATTERY is not part of the D-loop. Rather, it is wired directly to the auxiliary buttons labelled A through E, on the right-hand side of the Matrix panel. Note that no nixie display will occur unless one of the Battery buttons is depressed.

4.2 FADAC FIELD TABLE

The Field Table contains the computer power supply, which must be connected to a 3-phase, 400 HZ 120/208 volt source. The Field Table has 5 plugs labelled J-1 through J-5. J-5 is connected to the source voltage. J-1 through J-4 are identical, and are used for any peripheral equipment (such as the SDR or Teletypewriter) which requires the same source voltage. A cable, P-11, which is permanently fixed to the bottom of the Field Table behind J-4, is connected to J-11 on FADAC to energize the computer.

4.3 SIGNAL DATA REPRODUCER (SDR)

The Signal Data Reproducer (SDR) is the primary input device used with FADAC. It is a high-speed, photoelectric reader which operates at approximately 600 characters per second. Data input to the SDR may be 5 level Teletype, or 8 level Fieldata code. Figure 4.3 is a photograph of the SDR which shows the tape cannister, read head, operation controls, indicators, and SDR output plugs. To connect the SDR to FADAC, SDR - PA is connected to one of the power supply

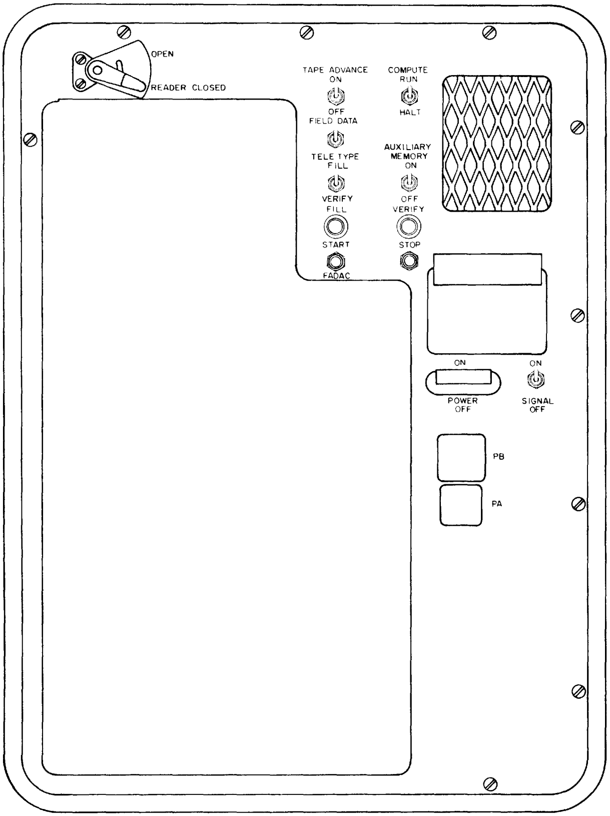


Figure 4-3. Signal Data Reproducer (SDR)

plugs (J-1 through J-4) on the Field Table.

The following is a description of the SDR controls and indicators:

Tape Advance On/Off.

Normally OFF, this switch is thrown to ON to cause the tape to mechanically advance to any desired position. No reading occurs while this switch is in the ON position.

Fieldata/Teletype.

The position of this switch determines whether information input to FADAC is interpreted as Fieldata or Teletype Code.

Fill/Verify Switch.

The position of this switch is immaterial, since it has no effect in either position. The switch was designed to control the input and verification of a continuous tape, in which the ends of the tape have been spliced together to form an endless loop.

Start FADAC.

This button is depressed to initiate Octal input to FADAC under external control. (Note that BCD or Alpha-numeric input is only possible under program control.)

Compute Run/Halt.

This switch is thrown into the HALT position when input is under external control; and into the RUN position when FADAC is under program control.

Auxiliary Memory Switch.

With this switch in the OFF position, all FADAC write heads are energized, i.e., all memory is HOT. With the switch in the ON position, only the write heads associated with HOT memory, as determined by the position of the HOT-Channel Select Switch, are energized. (See Section 4.5.)

Stop.

This button is depressed to manually stop the SDR tape.

Signal On-Off.

This switch is thrown to the ON position to energize the light source in the SDR Read head.

Power On/Off.

This switch is thrown to the ON position to energize the SDR.

Fill/Verify Indicators.

These neon bulbs are normally off. The appropriate one will light and remain on while information is input in the Fill or Verify mode.

Open/Close Handle.

This handle is OPEN to allow positioning of a paper tape; and in CLOSE position to lock the tape in place.

Figure 4.4 is a block diagram of a typical FADAC system, which illustrates how an input device (here the SDR), an output device, FADAC, and the Field Table are interconnected.

4.4 INPUT

General.

In most applications, paper tape is used to input information to FADAC. Either of two paper tape codes

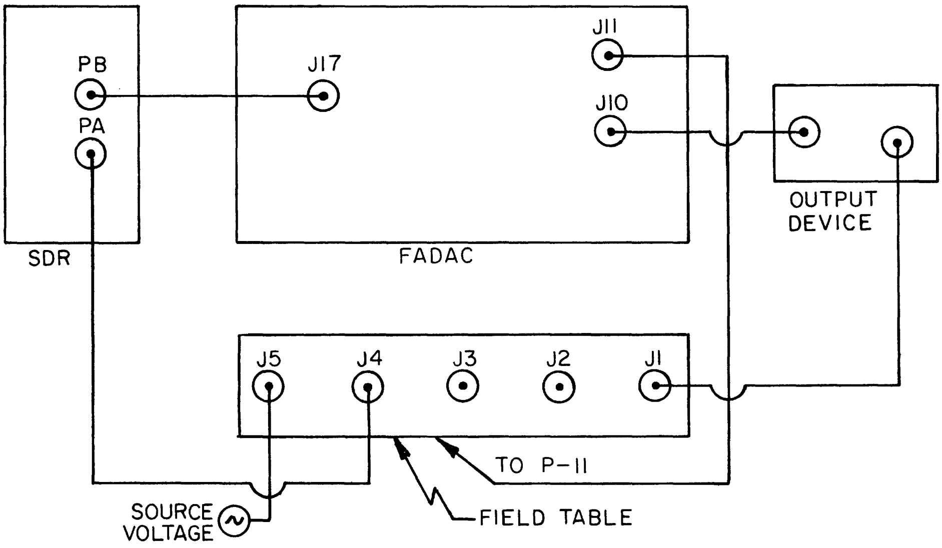


Figure 4-4. Block Diagram of a Typical FADAC System

may be used, 8-level Fieldata or 5-level Teletype code. When FADAC is under program control these input codes are interpreted and stored as Octal, BCD, Alpha-5 or Alpha-6, depending on the Read command sensed. Any input device which will generate Teletype or Fieldata paper tape codes may be used, whether or not paper tape is in fact the input vehicle. However, when FADAC is under external control, information can enter the computer in Octal format only.

Octal Input.

Recall that during octal input, control is with the input device, not with the computer itself. Therefore, in addition to the octal digits 0 through 7, FADAC will recognize seven special control characters. These characters are never stored in computer memory; rather each has a specific function. The blank, which can only be generated by a sprocket hole with no punches on paper tape, is ignored. (Paper tape codes for Octal, BCD digits and octal control characters are listed in Appendix C.)

4.4.1 OCTAL CONTROL CHARACTERS

Halt Code (8).

When this code is sensed, FADAC and the input device will halt. The I/O indicator light on the computer panel will turn off. If input is via the keyboard, the Keyboard Indicator light will also turn off.

Compute Code (9)

When this code is sensed, the Compute Indicator on the control panel will light, and computation will begin at the location specified by the 14 least significant bits in the L-register. (Note the L-register must be filled with the address of the first instruction to be executed prior to transmission of the Compute code.)

If the SDR is connected, computation will not start until the RUN/HALT switch is thrown into the RUN position.

Fill Code (RIGHT/UP/ADD/+).

This code may be used to put FADAC back into the FILL mode after it has been placed in the VERIFY mode.

Verify Code (LEFT/DROP/DOWN/–).

The VERIFY code causes the computer to compare the information following the VERIFY code with the information already stored in memory. If the data being [illegible] is not the same as that in memory, FADAC will [illegible] itself and the input device and the ERROR indicator [illegible] flash.

Location Code (.).

This code causes the 14 least significant bits in the A-register to be transferred to the L-register, thus setting up the initial location where the information following is to be stored, or where computation or verification is to start. Following a LOCATION code, each FADAC-Octal word entered into or verified by the computer causes the location in the L-register to be increased by one. This code can be given anywhere in a sequence of input data. The contents of the A-register are not changed by this control code.

Clear Code (CLEAR).

This code is ignored by FADAC during octal input.

Enter Code (ENTER).

This code will cause the contents of the A-register to be stored in the location specified by the 14 least significant bits in the L-register, after which the contents of L will be incremented by one. The contents of A remain unchanged.

Blank Code.

This code is ignored on octal input.

4.4.2 PREPARATION OF PAPER TAPES

FADAC paper tape is black, unoiled photo-electric tape, fan-folded into 8-1/2 inch folds. For teletype code, the tape is 11/16 inch wide; for Fieldata, one inch wide. Characters are punched 10 per inch. Any paper tape punch device may be used which will generate the paper tape codes listed in Appendix C and Appendix D. Note that the DELETE code (all ones) is used to cover a tape code which is punched in error on a program tape.

Octal Format.

All octal tapes should begin with a leader of approximately 2 folds of sprocket holes (17''), followed by a VERIFY code (–), followed by one fold of sprocket holes. The first data punched should be a 5-digit FADAC address, followed by a Location code (.), followed by an 11 digit FADAC-octal word, followed by an Enter Code. (Note: On the Friden and Soroban paper tape punch units the Enter Code is represented by the Carriage Return.) To shorten the length of a program tape, all information to be stored sequentially should be punched sequentially. This will eliminate the need to punch a 5-digit address for each entry. Each non-sequential entry MUST be preceded by the desired address followed by a Location code. The tape should end with a Halt code (8). If desired, however, the Halt code may be omitted and a 5-digit address, followed by the Location code, followed by the Compute code (9), may be the last punches on a tape. (If this latter method is used, computation will begin at the last location address specified, as soon as the SDR RUN/HALT switch is thrown into the RUN position). A leader of approximately two folds should follow the last character punched.

Non-Octal Tapes.

As stated, codes other than Octal are input under program control. The Read Command given will cause FADAC hardware to interpret and store the specified blocks of data in the desired format, beginning at the address specified in the L-register. The block size and type must be specified in all non-Octal Read Commands.

Tape Preparation - Non-Octal.

Non-octal tapes to be read into FADAC under program control are punched as an endless string of characters with no control characters, as such. Codes for alphanumeric input (Read Alpha-5 or Read Alpha-6) are listed in Appendix D, BCD in Appendix C.

Caution.

The SDR does not stop on a character. Rather approximately three spaces are lost each time an SDR stop is executed either manually or under program control. Therefore, if a tape is to be input via this device, it is essential that a space of at least 5 sprocket holes be left between each segment of information to be input by each Read External Device command.

4.5 PREPARATION OF THE FADAC SYSTEM

HOT Channel Select Switch.

This switch, located behind the Keyboard panel, is normally set for Hot 12 Channels. To change this setting, remove the 8 screws which fit the keyboard to the control panel. The Hot Channel Select switch is marked for 16 channels (left hand position), 4 channels (middle position), and 12 channels (right hand position), and may be manually set to the desired position.

SDR Settings.

With the SDR connected to FADAC, SDR controls are set as follows:

POWER - ON

SIGNAL - ON

TAPE ADVANCE - OFF

FD/TT - Set for input code desired.

FILL/VERIFY - Setting immaterial.

AUXILIARY MEMORY - OFF

COMPUTE RUN/HALT

Halt position for input under external control. In RUN position for all FADAC operations under program control.

4.5.1 STARTING THE SYSTEM

Turn on all peripheral equipment first.

Throw FADAC power switch to ON.

Do not perform any FADAC operations until the Power Ready indicator is on. This will occur approximately 20 seconds after power is set ON.

4.6. INPUT THROUGH EXTERNAL CONTROL (INPUT IN OCTAL)

4.6.1 PAPER TAPE (SDR)

To Insert Tape.

Check settings of the FD/TT switch and the tape guides at the SDR read head. To adjust the tape guides, turn screw head at each tape guide clockwise as far as possible to increase size for 8 level tape, and counter-clockwise to decrease size for 5 level tape. Position the OPEN/CLOSE handle to OPEN. Place tape through the tape guide slit located on the left side of the tape cartridge holder, and through the tape guides at the read head. The sprocket holes of the tape being between the 3rd and 4th levels of same, the tape should be positioned such that space for the 1st three levels is toward the inside wall of the SDR, Set the OPEN/CLOSE handle to CLOSE. (Check the tape guide settings by running a dummy tape through the SDR, as an improper setting may destroy a tape.)

Operation.

Press RESET.

Check setting of SDR controls. (Section 4.5.) Throw RUN/HALT switch to HALT.

Press RESET button on FADAC control panel, and START Button on SDR. The FILL indicator on the SDR and I/O indicator on FADAC will light, and remain lit throughout the tape input operation.

CAUTION.

Be sure the tape is positioned such that the Verify code is to the right of the read head.

To Verify.

Reposition the tape such that the VERIFY code is to the left of the read head. Press RESET on FADAC, and START on the SDR. The SDR VERIFY indicator and I/O indicator on FADAC will light. If an error is sensed, the ERROR indicator on FADAC will blink, and input will stop. Repeat tape input procedure described above and re-verify.

4.6.2 PAPER TAPE (MECHANICAL READER)

To Insert Tape.

Raise the clamp armature on the right hand side of the reader (Figure 4.5). The sprocket holes of the tape being between the 3rd and 4th levels of same, position the tape such that the space for the first three levels is toward the computer. Insure that all information to be input is to the left of the read head. Place the tape under the read head, engage the tape sprocket holes with the reader sprockets, and close the armature clamp. Turn the sprocket knob on the upper right side of the reader a few times to insure correct engagement. If the tape does not move freely, verify that the sprocket holes have made proper contact with the reader sprocket and that the tape is properly threaded between the read and the sprocket (Figure 4.5).

Operation.

Press RESET. Check setting of SDR controls. (See Section 4.5.) Throw RUN/HALT switch to HALT.

Press RESET and RECALL on FADAC, and START on the SDR. The FILL indicator on the SDR and the I/O indicator on FADAC will light, and remain lit throughout the tape input operation.

CAUTION.

Be sure the tape is positioned such that the VERIFY code is to the right of the read head.

To Verify.

Press RESET. Reposition the tape such that the

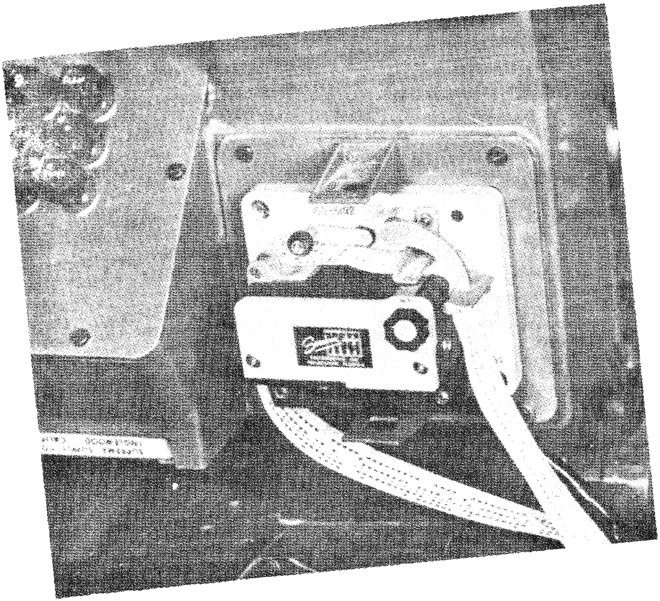


Figure 4.5. Mechanical Reader

VERIFY code is to the left of the read head. Press RESET and RECALL on FADAC and START on the SDR. The VERIFY indicator on the MLU and the I/O indicator on FADAC will light. If an error is sensed, the ERROR indicator on FADAC will blink, and input will stop. Repeat the tape input procedure described above and re-verify.

Keyboard Input.

Check setting of SDR controls. (See Section 4.5.) Throw RUN/HALT switch to HALT.

Press RESET and RECEIVE(or RESET and TRIG) on FADAC, and START on the SDR. The KEYBOARD and I/O indicators on FADAC and the FILL indicator on the SDR will light and remain on as long as FADAC remains in the Keyboard Input mode.

Type the 5-octal digit address to which the first entry is to be sent, followed by a LOCATION code (.), followed by the 11 octal digit word to be stored, followed by the ENTER code (ENTER key). If entries are to be sent to sequential locations, only the first assigned location followed by the LOCATION code must be typed. All subsequent entries may be 11 octal digits followed by the ENTER key. Each non-sequential entry MUST be preceded by the 5-digit assigned location followed by the LOCATION CODE.

Example:

Line 1	10125.10126250326E
Line 2	10126.30127010327E
Line 3	10127.10130511326E
Line 4	10130.14600150126E
Line 5	30126.00000372000E

Since the entries in lines 1 through 4 are sequential, only the first location field, 10125, must be typed. The contents of the L-register are automatically incremented by one each time an enter code (E) is sensed. To break the sequence, a new location field followed by a location code is entered, as in line 5 (30126). Note that this location in Memory III is identified by a 1 in bit position 18 of the L-register as follows:

	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	Bit Position
	0	1	1	0	0	0	0	0	1	0	1	0	1	1	0	Binary
L-Register	3			0			1			2			6			Octal

The format of the typed entries shown in the Example is precisely the same as that used to punch an octal paper tape.

To Verify Keyboard Entries.

Press LEFT DROP key on the keyboard and re-type all entries to be verified. If an error is detected, the ERROR indicator will blink. Re-enter the data, as described above and re-verify.

4.7 INPUT THROUGH PROGRAM CONTROL

(Note: Input may be in BCD, Alpha-5 or Alpha-6 format)

With the SDR Connected.

Check settings on the SDR. (See Section 4.5.) RUN/HALT switch must be in the RUN position. Check all peripheral equipment connections.

If Via SDR or Mechanical Reader.

Position tape properly in the input device and initiate computation. Tape will be input automatically when any Read command is sensed. The FILL indicator on the SDR and I/O indicator on FADAC will light, and remain on until the specified block size is entered.

If Via Keyboard.

Initiate computation. The Keyboard and I/O indicators on FADAC will light, and FADAC will halt to await a keyboard entry when any Read Keyboard command is sensed.

To Initiate Computation.

With the SDR (or SDR Simulator) Settings as specified in Section 4.5, the SDR RUN/HALT switch in the run position, and all peripheral equipment connected and operational, press the FADAC control button which is programmed to initiate computation.

APPENDIX A

Powers of Two Table

2^N	N	2^{-N}
1	0	1.0
2	1	0.5
4	2	0.25
8	3	0.125
16	4	0.062 5
32	5	0.031 25
64	6	0.015 625
128	7	0.007 812 5
256	8	0.003 906 25
512	9	0.001 953 125
1 024	10	0.000 976 562 5
2 048	11	0.000 488 281 25
4 096	12	0.000 244 140 625
8 192	13	0.000 122 070 312 5
16 384	14	0.000 061 035 156 25
32 768	15	0.000 030 517 578 125
65 536	16	0.000 015 258 789 062 5
131 072	17	0.000 007 629 394 531 25
262 144	18	0.000 003 814 697 265 625
524 288	19	0.000 001 907 348 632 812 5
1 048 576	20	0.000 000 953 674 316 406 25
2 097 152	21	0.000 000 476 837 158 203 125
4 194 304	22	0.000 000 238 418 579 101 562 5
8 388 608	23	0.000 000 119 209 289 550 781 25
16 777 216	24	0.000 000 059 604 644 775 390 625
33 554 432	25	0.000 000 029 802 322 387 695 312 5
67 108 864	26	0.000 000 014 901 161 193 847 656 25
134 217 728	27	0.000 000 007 450 580 596 923 828 125
268 435 456	28	0.000 000 003 725 290 298 461 914 062 5
536 870 912	29	0.000 000 001 862 645 149 230 957 031 25
1 073 741 824	30	0.000 000 000 931 322 574 615 478 515 625
2 147 483 648	31	0.000 000 000 465 661 287 307 739 257 812 5

APPENDIX B

OCTAL-DECIMAL INTEGER CONVERSION TABLE

0000 to 0777 (Octal)	0000 to 0511 (Decimal)

	0	1	2	3	4	5	6	7
0000	0000	0001	0002	0003	0004	0005	0006	0007
0010	0008	0009	0010	0011	0012	0013	0014	0015
0020	0016	0017	0018	0019	0020	0021	0022	0023
0030	0024	0025	0026	0027	0028	0029	0030	0031
0040	0032	0033	0034	0035	0036	0037	0038	0039
0050	0040	0041	0042	0043	0044	0045	0046	0047
0060	0048	0049	0050	0051	0052	0053	0054	0055
0070	0056	0057	0058	0059	0060	0061	0062	0063
0100	0064	0065	0066	0067	0068	0069	0070	0071
0110	0072	0073	0074	0075	0076	0077	0078	0079
0120	0080	0081	0082	0083	0084	0085	0086	0087
0130	0088	0089	0090	0091	0092	0093	0094	0095
0140	0096	0097	0098	0099	0100	0101	0102	0103
0150	0104	0105	0106	0107	0108	0109	0110	0111
0160	0112	0113	0114	0115	0116	0117	0118	0119
0170	0120	0121	0122	0123	0124	0125	0126	0127
0200	0128	0129	0130	0131	0132	0133	0134	0135
0210	0136	0137	0138	0139	0140	0141	0142	0143
0220	0144	0145	0146	0147	0148	0149	0150	0151
0230	0152	0153	0154	0155	0156	0157	0158	0159
0240	0160	0161	0162	0163	0164	0165	0166	0167
0250	0168	0169	0170	0171	0172	0173	0174	0175
0260	0176	0177	0178	0179	0180	0181	0182	0183
0270	0184	0185	0186	0187	0188	0189	0190	0191
0300	0192	0193	0194	0195	0196	0197	0198	0199
0310	0200	0201	0202	0203	0204	0205	0206	0207
0320	0208	0209	0210	0211	0212	0213	0214	0215
0330	0216	0217	0218	0219	0220	0221	0222	0223
0340	0224	0225	0226	0227	0228	0229	0230	0231
0350	0232	0233	0234	0235	0236	0237	0238	0239
0360	0240	0241	0242	0243	0244	0245	0246	0247
0370	0248	0249	0250	0251	0252	0253	0254	0255

	0	1	2	3	4	5	6	7
0400	0256	0257	0258	0259	0260	0261	0262	0263
0410	0264	0265	0266	0267	0268	0269	0270	0271
0420	0272	0273	0274	0275	0276	0277	0278	0279
0430	0280	0281	0282	0283	0284	0285	0286	0287
0440	0288	0289	0290	0291	0292	0293	0294	0295
0450	0296	0297	0298	0299	0300	0301	0302	0303
0460	0304	0305	0306	0307	0308	0309	0310	0311
0470	0312	0313	0314	0315	0316	0317	0318	0319
0500	0320	0321	0322	0323	0324	0325	0326	0327
0510	0328	0329	0330	0331	0332	0333	0334	0335
0520	0336	0337	0338	0339	0340	0341	0342	0343
0530	0344	0345	0346	0347	0348	0349	0350	0351
0540	0352	0353	0354	0355	0356	0357	0358	0359
0550	0360	0361	0362	0363	0364	0365	0366	0367
0560	0368	0369	0370	0371	0372	0373	0374	0375
0570	0376	0377	0378	0379	0380	0381	0382	0383
0600	0384	0385	0386	0387	0388	0389	0390	0391
0610	0392	0393	0394	0395	0396	0397	0398	0399
0620	0400	0401	0402	0403	0404	0405	0406	0407
0630	0408	0409	0410	0411	0412	0413	0414	0415
0640	0416	0417	0418	0419	0420	0421	0422	0423
0650	0424	0425	0426	0427	0428	0429	0430	0431
0660	0432	0433	0434	0435	0436	0437	0438	0439
0670	0440	0441	0442	0443	0444	0445	0446	0447
0700	0448	0449	0450	0451	0452	0453	0454	0455
0710	0456	0457	0458	0459	0460	0461	0462	0463
0720	0464	0465	0466	0467	0468	0469	0470	0471
0730	0472	0473	0474	0475	0476	0477	0478	0479
0740	0480	0481	0482	0483	0484	0485	0486	0487
0750	0488	0489	0490	0491	0492	0493	0494	0495
0760	0496	0497	0498	0499	0500	0501	0502	0503
0770	0504	0505	0506	0507	0508	0509	0510	0511

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

1000 to 1777 (Octal)	0512 to 1023 (Decimal)

	0	1	2	3	4	5	6	7
1000	0512	0513	0514	0515	0516	0517	0518	0519
1010	0520	0521	0522	0523	0524	0525	0526	0527
1020	0528	0529	0530	0531	0532	0533	0534	0535
1030	0536	0537	0538	0539	0540	0541	0542	0543
1040	0544	0545	0546	0547	0548	0549	0550	0551
1050	0552	0553	0554	0555	0556	0557	0558	0559
1060	0560	0561	0562	0563	0564	0565	0566	0567
1070	0568	0569	0570	0571	0572	0573	0574	0575
1100	0576	0577	0578	0579	0580	0581	0582	0583
1110	0584	0585	0586	0587	0588	0589	0590	0591
1120	0592	0593	0594	0595	0596	0597	0598	0599
1130	0600	0601	0602	0603	0604	0605	0606	0607
1140	0608	0609	0610	0611	0617	0813	0614	0615
1150	0616	0617	0618	0619	0620	0621	0622	0623
1160	0624	0625	0626	0627	0628	0629	0630	0631
1170	0632	0633	0634	0635	0636	0637	0638	0639
1200	0640	0641	0642	0643	0644	0645	0646	0647
1210	0648	0649	0650	0651	0652	0653	0654	0655
1220	0656	0657	0658	0659	0660	0661	0662	0663
1230	0664	0665	0666	0667	0668	0669	0670	0671
1240	0672	0673	0674	0675	0676	0677	0678	0679
1250	0680	0681	0682	0683	0684	0685	0686	0687
1260	0688	0689	0690	0691	0692	0693	0694	0695
1270	0696	0697	0698	0699	0700	0701	0702	0703
1300	0704	0705	0706	0707	0708	0709	0710	0711
1310	0712	0713	0714	0715	0716	0717	0718	0719
1320	0720	0721	0722	0723	0724	0725	0726	0727
1330	0728	0729	0730	0731	0732	0733	0734	0735
1340	0736	0737	0738	0739	0740	0741	0742	0743
1350	0744	0745	0746	0747	0748	0749	0750	0751
1360	0752	0753	0754	0755	0756	0757	0758	0759
1370	0760	0761	0762	0763	0764	0765	0766	0767

	0	1	2	3	4	5	6	7
1400	0768	0769	0770	0771	0772	0773	0774	0775
1410	0776	0777	0778	0779	0780	0781	0782	0783
1420	0784	0785	0786	0787	0788	0789	0790	0791
1430	0792	0793	0794	0795	0796	0797	0798	0799
1440	0800	0801	0802	0803	0804	0805	0806	0807
1450	0808	0809	0810	0811	0812	0813	0814	0815
1460	0816	0817	0818	0819	0820	0821	0822	0823
1470	0824	0825	0826	0827	0828	0829	0830	0831
1500	0832	0833	0834	0835	0836	0837	0838	0839
1510	0840	0841	0842	0843	0844	0845	0846	0847
1520	0848	0849	0850	0851	0852	0853	0854	0855
1530	0856	0857	0858	0859	0860	0861	0862	0863
1540	0864	0865	0866	0867	0868	0869	0870	0871
1550	0872	0873	0874	0875	0876	0877	0878	0879
1560	0880	0881	0882	0883	0884	0885	0886	0887
1570	0888	0889	0890	0891	0892	0893	0894	0895
1600	0896	0897	0898	0899	0900	0901	0902	0903
1610	0904	0905	0906	0907	0908	0909	0910	0911
1620	0912	0913	0914	0915	0916	0917	0918	0919
1630	0920	0921	0922	0923	0924	0925	0926	0927
1640	0928	0929	0930	0931	0932	0933	0934	0935
1650	0936	0937	0938	0939	0940	0941	0942	0943
1660	0944	0945	0946	0947	0948	0949	0950	0951
1670	0952	0953	0954	0955	0956	0957	0958	0959
1700	0960	0961	0962	0963	0964	0965	0966	0967
1710	0968	0969	0970	0971	0972	0973	0974	0975
1720	0976	0977	0978	0979	0980	0981	0982	0983
1730	0984	0985	0986	0987	0988	0989	0990	0991
1740	0992	0993	0994	0995	0996	0997	0998	0999
1750	1000	1001	1002	1003	1004	1005	1006	1007
1760	1008	1009	1010	1011	1012	1013	1014	1015
1770	1016	1017	1018	1019	1020	1021	1022	1023

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

2000 to 2777 (Octal)	1024 to 1535 (Decimal)

	0	1	2	3	4	5	6	7
2000	1024	1025	1026	1027	1028	1029	1030	1031
2010	1032	1033	1034	1035	1036	1037	1038	1039
2020	1040	1041	1042	1043	1044	1045	1046	1047
2030	1048	1049	1050	1051	1052	1053	1054	1055
2040	1056	1057	1058	1059	1060	1061	1062	1063
2050	1064	1065	1066	1067	1068	1069	1070	1071
2060	1072	1073	1074	1075	1076	1077	1078	1079
2070	1080	1081	1082	1083	1084	1085	1086	1087
2100	1088	1089	1090	1091	1092	1093	1094	1095
2110	1096	1097	1098	1099	1100	1101	1102	1103
2120	1104	1105	1106	1107	1108	1109	1110	1111
2130	1112	1113	1114	1115	1116	1117	1118	1119
2140	1120	1121	1122	1123	1124	1125	1126	1127
2150	1128	1129	1130	1131	1132	1133	1134	1135
2160	1136	1137	1138	1139	1140	1141	1142	1143
2170	1144	1145	1146	1147	1148	1149	1150	1151
2200	1152	1153	1154	1155	1156	1157	1158	1159
2210	1160	1161	1162	1163	1164	1165	1166	1167
2220	1168	1169	1170	1171	1172	1173	1174	1175
2230	1176	1177	1178	1179	1180	1181	1182	1183
2240	1184	1185	1186	1187	1188	1189	1190	1191
2250	1192	1193	1194	1195	1196	1197	1198	1199
2260	1200	1201	1202	1203	1204	1205	1206	1207
2270	1208	1209	1210	1211	1212	1213	1214	1215
2300	1216	1217	1218	1219	1220	1221	1222	1223
2310	1224	1225	1226	1227	1228	1229	1230	1231
2320	1232	1233	1234	1235	1236	1237	1238	1239
2330	1240	1241	1242	1243	1244	1245	1246	1247
2340	1248	1249	1250	1251	1252	1253	1254	1255
2350	1256	1257	1258	1259	1260	1261	1262	1263
2360	1264	1265	1266	1267	1268	1269	1270	1271
2370	1272	1273	1274	1275	1276	1277	1278	1279
2400	1280	1281	1282	1283	1284	1285	1286	1287
2410	1288	1289	1290	1291	1292	1293	1294	1295
2420	1296	1297	1298	1299	1300	1301	1302	1303
2430	1304	1305	1306	1307	1308	1309	1310	1311
2440	1312	1313	1314	1315	1316	1317	1318	1319
2450	1320	1321	1322	1323	1324	1325	1326	1327
2460	1328	1329	1330	1331	1332	1333	1334	1335
2470	1336	1337	1338	1339	1340	1341	1342	1343
2500	1344	1345	1346	1347	1348	1349	1350	1351
2510	1352	1353	1354	1355	1356	1357	1358	1359
2520	1360	1361	1362	1363	1364	1365	1366	1367
2530	1368	1369	1370	1371	1372	1373	1374	1375
2540	1376	1377	1378	1379	1380	1381	1382	1383
2550	1384	1385	1386	1387	1388	1389	1390	1391
2560	1392	1393	1394	1395	1396	1397	1398	1399
2570	1400	1401	1402	1403	1404	1405	1406	1407
2600	1408	1409	1410	1411	1412	1413	1414	1415
2610	1416	1417	1418	1419	1420	1421	1422	1423
2620	1424	1425	1426	1427	1428	1429	1430	1431
2630	1432	1433	1434	1435	1436	1437	1438	1439
2640	1440	1441	1442	1443	1444	1445	1446	1447
2650	1448	1449	1450	1451	1452	1453	1454	1455
2660	1456	1457	1458	1459	1460	1461	1462	1463
2670	1464	1465	1466	1467	1468	1469	1470	1471
2700	1472	1473	1474	1475	1476	1477	1478	1479
2710	1480	1481	1482	1483	1484	1485	1486	1487
2720	1488	1489	1490	1491	1492	1493	1494	1495
2730	1496	1497	1498	1499	1500	1501	1502	1503
2740	1504	1505	1506	1507	1508	1509	1510	1511
2750	1512	1513	1514	1515	1516	1517	1518	1519
2760	1520	1521	1522	1523	1524	1525	1526	1527
2770	1528	1529	1530	1531	1532	1533	1534	1535

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont d)

3000	1536
to	to
3777	2047
(Octal)	(Decimal)

	0	1	2	3	4	5	6	7
3000	1536	1537	1538	1539	1540	1541	1542	1543
3010	1544	1545	1546	1547	1548	1549	1550	1551
3020	1552	1553	1554	1555	1556	1557	1558	1559
3030	1560	1561	1562	1563	1564	1565	1566	1567
3040	1568	1569	1570	1571	1572	1573	1574	1575
3050	1576	1577	1578	1579	1580	1581	1582	1583
3060	1584	1585	1586	1587	1588	1589	1590	1591
3070	1592	1593	1594	1595	1596	1597	1598	1599
3100	1600	1601	1602	1603	1604	1605	1606	1607
3110	1608	1609	1610	1611	1612	1613	1614	1615
3120	1616	1617	1618	1619	1620	1621	1622	1623
3130	1624	1625	1626	1627	1628	1629	1630	1631
3140	1632	1633	1634	1635	1636	1637	1638	1639
3150	1640	1641	1642	1643	1644	1645	1646	1647
3160	1648	1649	1650	1651	1652	1653	1654	1655
3170	1656	1657	1658	1659	1660	1661	1662	1663
3200	1664	1665	1666	1667	1668	1669	1670	1671
3210	1672	1673	1674	1675	1676	1677	1678	1679
3220	1680	1681	1682	1683	1684	1685	1686	1687
3230	1688	1689	1690	1691	1692	1693	1694	1695
3240	1696	1697	1698	1699	1700	1701	1702	1703
3250	1704	1705	1706	1707	1708	1709	1710	1711
3260	1712	1713	1714	1715	1716	1717	1718	1719
3270	1720	1721	1722	1723	1724	1725	1726	1727
3300	1728	1729	1730	1731	1732	1733	1734	1735
3310	1736	1737	1738	1739	1740	1741	1742	1743
3320	1744	1745	1746	1747	1748	1749	1750	1751
3330	1752	1753	1754	1755	1756	1757	1758	1759
3340	1760	1761	1762	1763	1764	1765	1766	1767
3350	1768	1769	1770	1771	1772	1773	1774	1775
3360	1776	1777	1778	1779	1780	1781	1782	1783
3370	1784	1785	1786	1787	1788	1789	1790	1791

	0	1	2	3	4	5	6	7
3400	1792	1793	1794	1795	1796	1797	1798	1799
3410	1800	1801	1802	1803	1804	1805	1806	1807
3420	1808	1809	1810	1811	1812	1813	1814	1815
3430	1816	1817	1818	1819	1820	1821	1822	1823
3440	1824	1825	1826	1827	1828	1829	1830	1831
3450	1832	1833	1834	1835	1836	1837	1838	1839
3460	1840	1841	1842	1843	1844	1845	1846	1847
3470	1848	1849	1850	1851	1852	1853	1854	1855
3500	1856	1857	1858	1859	1860	1861	1862	1863
3510	1864	1865	1866	1867	1868	1869	1870	1871
3520	1872	1873	1874	1875	1876	1877	1878	1879
3530	1880	1881	1882	1883	1884	1885	1886	1887
3540	1888	1889	1890	1891	1892	1893	1894	1895
3550	1896	1897	1898	1899	1900	1901	1902	1903
3560	1904	1905	1906	1907	1908	1909	1910	1911
3570	1912	1913	1914	1915	1916	1917	1918	1919
3600	1920	1921	1922	1923	1924	1925	1926	1927
3610	1928	1929	1930	1931	1932	1933	1934	1935
3620	1936	1937	1938	1939	1940	1941	1942	1943
3630	1944	1945	1946	1947	1948	1949	1950	1951
3640	1952	1953	1954	1955	1956	1957	1958	1959
3650	1960	1961	1962	1963	1964	1965	1966	1967
3660	1968	1969	1970	1971	1972	1973	1974	1975
3670	1976	1977	1978	1979	1980	1981	1982	1983
3700	1984	1985	1986	1987	1988	1989	1990	1991
3710	1992	1993	1994	1995	1996	1997	1998	1999
3720	2000	2001	2002	2003	2004	2005	2006	2007
3730	2008	2009	2010	2011	2012	2013	2014	2015
3740	2016	2017	2018	2019	2020	2021	2022	2023
3750	2024	2025	2026	2027	2028	2029	2030	2031
3760	2032	2033	2034	2035	2036	2037	2038	2039
3770	2040	2041	2042	2043	2044	2045	2046	2047

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

4000	2048
to	to
4777	2559
(Octal)	(Decimal)

	0	1	2	3	4	5	6	7
4000	2048	2049	2050	2051	2052	2053	2054	2055
4010	2056	2057	2058	2059	2060	2061	2062	2063
4020	2064	2065	2066	2067	2068	2069	2070	2071
4030	2072	2073	2074	2075	2076	2077	2078	2079
4040	2080	2081	2082	2083	2084	2085	2086	2087
4050	2088	2089	2090	2091	2092	2093	2094	2095
4060	2096	2097	2098	2099	2100	2101	2102	2103
4070	2104	2105	2106	2107	2108	2109	2110	2111
4100	2112	2113	2114	2115	2116	2117	2118	2119
4110	2120	2121	2122	2123	2124	2125	2126	2127
4120	2128	2129	2130	2131	2132	2133	2134	2135
4130	2136	2137	2138	2139	2140	2141	2142	2143
4140	2144	2145	2146	2147	2148	2149	2150	2151
4150	2152	2153	2154	2155	2156	2157	2158	2159
4160	2160	2161	2162	2163	2164	2165	2166	2167
4170	2168	2169	2170	2171	2172	2173	2174	2175
4200	2176	2177	2178	2179	2180	2181	2182	2183
4210	2184	2185	2186	2187	2188	2189	2190	2191
4220	2192	2193	2194	2195	2196	2197	2198	2199
4230	2200	2201	2202	2203	2204	2205	2206	2207
4240	2208	2209	2210	2211	2212	2213	2214	2215
4250	2216	2217	2218	2219	2220	2221	2222	2223
4260	2224	2225	2226	2227	2228	2229	2230	2231
4270	2232	2233	2234	2235	2236	2237	2238	2239
4300	2240	2241	2242	2243	2244	2245	2246	2247
4310	2248	2249	2250	2251	2252	2253	2254	2255
4320	2256	2257	2258	2259	2260	2261	2262	2263
4330	2264	2265	2266	2267	2268	2269	2270	2271
4340	2272	2273	2274	2275	2276	2277	2278	2279
4350	2280	2281	2282	2283	2284	2285	2286	2287
4360	2288	2289	2290	2291	2292	2293	2294	2295
4370	2296	2297	2298	2299	2300	2301	2302	2303

	0	1	2	3	4	5	6	7
4400	2304	2305	2306	2307	2308	2309	2310	2311
4410	2312	2313	2314	2315	2316	2317	2318	2319
4420	2320	2321	2322	2323	2324	2325	2326	2327
4430	2328	2329	2330	2331	2332	2333	2334	2335
4440	2336	2337	2338	2339	2340	2341	2342	2343
4450	2344	2345	2346	2347	2348	2349	2350	2351
4460	2352	2353	2354	2355	2356	2357	2358	2359
4470	2360	2361	2362	2363	2364	2365	2366	2367
4500	2368	2369	2370	2371	2372	2373	2374	2375
4510	2376	2377	2378	2379	2380	2381	2382	2383
4520	2384	2385	2386	2387	2388	2389	2390	2391
4530	2392	2393	2394	2395	2396	2397	2398	2399
4540	2400	2401	2402	2403	2404	2405	2406	2407
4550	2408	2409	2410	2411	2412	2413	2414	2415
4560	2416	2417	2418	2419	2420	2421	2422	2423
4570	2424	2425	2426	2427	2428	2429	2430	2431
4600	2432	2433	2434	2435	2436	2437	2438	2439
4610	2440	2441	2442	2443	2444	2445	2446	2447
4620	2448	2449	2450	2451	2452	2453	2454	2455
4630	2456	2457	2458	2459	2460	2461	2462	2463
4640	2464	2465	2466	2467	2468	2469	2470	2471
4650	2472	2473	2474	2475	2476	2477	2478	2479
4660	2480	2481	2482	2483	2484	2485	2486	2487
4670	2488	2489	2490	2491	2492	2493	2494	2495
4700	2496	2497	2498	2499	2500	2501	2502	2503
4710	2504	2505	2506	2507	2508	2509	2510	2511
4720	2512	2513	2514	2515	2516	2517	2518	2519
4730	2520	2521	2522	2523	2524	2525	2526	2527
4740	2528	2529	2530	2531	2532	2533	2534	2535
4750	2536	2537	2538	2539	2540	2541	2542	2543
4760	2544	2545	2546	2547	2548	2549	2550	2551
4770	2552	2553	2554	2555	2556	2557	2558	2559

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

5000	2560
to	to
5777	3071
(Octal)	(Decimal)

	0	1	2	3	4	5	6	7
5000	2560	2561	2562	2563	2564	2565	2566	2567
5010	2568	2569	2570	2571	2572	2573	2574	2575
5020	2576	2577	2578	2579	2580	2581	2582	2583
5030	2584	2585	2586	2587	2588	2589	2590	2591
5040	2592	2593	2594	2595	2596	2597	2598	2599
5050	2600	2601	2602	2603	2604	2605	2606	2607
5060	2608	2609	2610	2611	2612	2613	2614	2615
5070	2616	2617	2618	2619	2620	2621	2622	2623
5100	2624	2625	2626	2627	2628	2629	2630	2631
5110	2632	2633	2634	2635	2636	2637	2638	2639
5120	2640	2641	2642	2643	2644	2645	2646	2647
5130	2648	2649	2650	2651	2652	2653	2654	2655
5140	2656	2657	2658	2659	2660	2661	2662	2663
5150	2664	2665	2666	2667	2668	2669	2670	2671
5160	2672	2673	2674	2675	2676	2677	2678	2679
5170	2680	2681	2682	2683	2684	2685	2686	2687
5200	2688	2689	2690	2691	2692	2693	2694	2695
5210	2696	2697	2698	2699	2700	2701	2702	2703
5220	2704	2705	2706	2707	2708	2709	2710	2711
5230	2712	2713	2714	2715	2716	2717	2718	2719
5240	2720	2721	2722	2723	2724	2725	2726	2727
5250	2728	2729	2730	2731	2732	2733	2734	2735
5260	2736	2737	2738	2739	2740	2741	2742	2743
5270	2744	2745	2746	2747	2748	2749	2750	2751
5300	2752	2753	2754	2755	2756	2757	2758	2759
5310	2760	2761	2762	2763	2764	2765	2766	2767
5320	2768	2769	2770	2771	2772	2773	2774	2775
5330	2776	2777	2778	2779	2780	2781	2782	2783
5340	2784	2785	2786	2787	2788	2789	2790	2791
5350	2792	2793	2794	2795	2796	2797	2798	2799
5360	2800	2801	2802	2803	2804	2805	2806	2807
5370	2808	2809	2810	2811	2812	2813	2814	2815

	0	1	2	3	4	5	6	7
5400	2816	2817	2818	2819	2820	2821	2822	2823
5410	2824	2825	2826	2827	2828	2829	2830	2831
5420	2832	2833	2834	2835	2836	2837	2838	2839
5430	2840	2841	2842	2843	2844	2845	2846	2847
5440	2848	2849	2850	2851	2852	2853	2854	2855
5450	2856	2857	2858	2859	2860	2861	2862	2863
5460	2864	2865	2866	2867	2868	2869	2870	2871
5470	2872	2873	2874	2875	2876	2877	2878	2879
5500	2880	2881	2882	2883	2884	2885	2886	2887
5510	2888	2889	2890	2891	2892	2893	2894	2895
5520	2896	2897	2898	2899	2900	2901	2902	2903
5530	2904	2905	2906	2907	2908	2909	2910	2911
5540	2912	2913	2914	2915	2916	2917	2918	2919
5550	2920	2921	2922	2923	2924	2925	2926	2927
5560	2928	2929	2930	2931	2932	2933	2934	2935
5570	2936	2937	2938	2939	2940	2941	2942	2943
5600	2944	2945	2946	2947	2948	2949	2950	2951
5610	2952	2953	2954	2955	2956	2957	2958	2959
5620	2960	2961	2962	2963	2964	2965	2966	2967
5630	2968	2969	2970	2971	2972	2973	2974	2975
5640	2976	2977	2978	2979	2980	2981	2982	2983
5650	2984	2985	2986	2987	2988	2989	2990	2991
5660	2992	2993	2994	2995	2996	2997	2998	2999
5670	3000	3001	3002	3003	3004	3005	3006	3007
5700	3008	3009	3010	3011	3012	3013	3014	3015
5710	3016	3017	3018	3019	3020	3021	3022	3023
5720	3024	3025	3026	3027	3028	3029	3030	3031
5730	3032	3033	3034	3035	3036	3037	3038	3039
5740	3040	3041	3042	3043	3044	3045	3046	3047
5750	3048	3049	3050	3051	3052	3053	3054	3055
5760	3056	3057	3058	3059	3060	3061	3062	3063
5770	3064	3065	3066	3067	3068	3069	3070	3071

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

6000	3072
to	to
6777	3583
(Octal)	(Decimal)

	0	1	2	3	4	5	6	7
6000	3072	3073	3074	3075	3076	3077	3078	3079
6010	3080	3081	3082	3083	3084	3085	3086	3087
6020	3088	3089	3090	3091	3092	3093	3094	3095
6030	3096	3097	3098	3099	3100	3101	3102	3103
6040	3104	3105	3106	3107	3108	3109	3110	3111
6050	3112	3113	3114	3115	3116	3117	3118	3119
6060	3120	3121	3122	3123	3124	3125	3126	3127
6070	3128	3129	3130	3131	3132	3133	3134	3135
6100	3136	3137	3138	3139	3140	3141	3142	3143
6110	3144	3145	3146	3147	3148	3149	3150	3151
6120	3152	3153	3154	3155	3156	3157	3158	3159
6130	3160	3161	3162	3163	3164	3165	3166	3167
6140	3168	3169	3170	3171	3172	3173	3174	3175
6150	3176	3177	3178	3179	3180	3181	3182	3183
6160	3184	3185	3186	3187	3188	3189	3190	3191
6170	3192	3193	3194	3195	3196	3197	3198	3199
6200	3200	3201	3202	3203	3204	3205	3206	3207
6210	3208	3209	3210	3211	3212	3213	3214	3215
6220	3216	3217	3218	3219	3220	3221	3222	3223
6230	3224	3225	3226	3227	3228	3229	3230	3231
6240	3232	3233	3234	3235	3236	3237	3238	3239
6250	3240	3241	3242	3243	3244	3245	3246	3247
6260	3248	3249	3250	3251	3252	3253	3254	3255
6270	3256	3257	3258	3259	3260	3261	3262	3263
6300	3264	3265	3266	3267	3268	3269	3270	3271
6310	3272	3273	3274	3275	3276	3277	3278	3279
6320	3280	3281	3282	3283	3284	3285	3286	3287
6330	3288	3289	3290	3201	3292	3293	3204	3295
6340	3296	3297	3298	3299	3300	3301	3302	3303
6350	3304	3305	3306	3307	3303	3309	3310	3311
6360	3312	3313	3314	3315	3316	3317	3318	3319
6370	3320	3321	3322	3323	3324	3325	3326	3327

	0	1	2	3	4	5	6	7
6400	3328	3329	3330	3331	3332	3333	3334	3335
6410	3336	3337	3338	3339	3340	3341	3342	3343
6420	3344	3345	3346	3347	3348	3349	3350	3351
6430	3352	3353	3354	3355	3356	3357	3358	3359
6440	3360	3361	3362	3363	3364	3365	3366	3367
6450	3368	3369	3370	3371	3372	3373	3374	3375
6460	3376	3377	3378	3379	3380	3381	3382	3383
6470	3384	3385	3386	3387	3388	3389	3390	3391
6500	3392	3393	3394	3395	3396	3397	3398	3399
6510	3400	3401	3402	3403	3404	3405	3406	3407
6520	3408	3409	3410	3411	3412	3413	3414	3415
6530	3416	3417	3418	3419	3420	3421	3422	3423
6540	3424	3425	3426	3427	3428	3429	3430	3431
6550	3432	3433	3434	3435	3436	3437	3438	3439
6560	3440	3441	3442	3443	3444	3445	3446	3447
6570	3448	3449	3450	3451	3452	3453	3454	3455
6600	3456	3457	3458	3459	3460	3461	3462	3463
6610	3464	3465	3466	3467	3468	3469	3470	3471
6620	3472	3473	3474	3475	3476	3477	3478	3479
6630	3480	3481	3482	3483	3484	3485	3486	3487
6640	3488	3489	3490	3491	3492	3493	3494	3495
6650	3496	3497	3498	3499	3500	3501	3502	3503
6660	3504	3505	3506	3507	3508	3509	3510	3511
6670	3512	3513	3514	3515	3516	3517	3518	3519
6700	3520	3521	3522	3523	3524	3525	3526	3527
6710	3528	3529	3530	3531	3532	3533	3534	3535
6720	3536	3537	3538	3539	3540	3541	3542	3543
6730	3544	3545	3546	3547	3548	3549	3550	3551
6740	3552	3553	3554	3555	3556	3557	3558	3559
6750	3560	3561	3562	3563	3564	3565	3566	3567
6760	3568	3569	3570	3571	3572	3573	3574	3575
6770	3576	3577	3578	3579	3580	3581	3582	3583

OCTAL-DECIMAL INTEGER CONVERSION TABLE (Cont'd)

7000	3584
to	to
7777	4095
(Octal)	(Decimal)

	0	1	2	3	4	5	6	7
7000	3584	3585	3586	3587	3588	3589	3590	3591
7010	3592	3593	3594	3595	3596	3597	3598	3599
7020	3600	3601	3602	3603	3604	3605	3606	3607
7030	3608	3609	3610	3611	3612	3613	3614	3615
7040	3616	3617	3618	3619	3620	3621	3622	3623
7050	3624	3625	3626	3627	3628	3629	3630	3631
7060	3632	3633	3634	3635	3636	3637	3638	3639
7070	3640	3641	3642	3643	3644	3645	3646	3647
7100	3648	3649	3650	3651	3652	3653	3654	3655
7110	3656	3657	3658	3659	3660	3661	3662	3663
7120	3664	3665	3666	3667	3668	3669	3670	3671
7130	3672	3673	3674	3675	3676	3677	3678	3679
7140	3680	3681	3682	3683	3684	3685	3686	3687
7150	3688	3689	3690	3691	3692	3693	3694	3695
7160	3696	3697	3698	3699	3700	3701	3702	3703
7170	3704	3705	3706	3707	3708	3709	3710	3711
7200	3712	3713	3714	3715	3716	3717	3718	3719
7210	3720	3721	3722	3723	3724	3725	3726	3727
7220	3728	3729	3730	3731	3732	3733	3734	3735
7230	3736	3737	3738	3739	3740	3741	3742	3743
7240	3744	3745	3746	3747	3748	3749	3750	3751
7250	3752	3753	3754	3755	3756	3757	3758	3759
7260	3760	3761	3762	3763	3764	3765	3766	3767
7270	3768	3769	3770	3771	3772	3773	3774	3775
7300	3776	3777	3778	3779	3780	3781	3782	3783
7310	3784	3785	3786	3787	3788	3789	3790	3791
7320	3792	3793	3794	3795	3796	3797	3798	3799
7330	3800	3801	3802	3803	3804	3805	3806	3807
7340	3808	3809	3810	3811	3812	3813	3814	3815
7350	3816	3817	3818	3819	3820	3821	3822	3823
7360	3824	3825	3826	3827	3828	3829	3830	3831
7370	3832	3833	3834	3835	3836	3837	3838	3839

	0	1	2	3	4	5	6	7
7400	3840	3841	3842	3843	3844	3845	3846	3847
7410	3848	3849	3850	3851	3852	3853	3854	3855
7420	3856	3857	3858	3859	3860	3861	3862	3863
7430	3864	3865	3866	3867	3868	3869	3870	3871
7440	3872	3873	3874	3875	3876	3877	3878	3879
7450	3880	3881	3882	3883	3884	3885	3886	3887
7460	3888	3889	3890	3891	3892	3893	3894	3895
7470	3896	3897	3898	3899	3900	3901	3902	3903
7500	3904	3905	3906	3907	3908	3909	3910	3911
7510	3912	3913	3914	3915	3916	3917	3918	3919
7520	3920	3921	3922	3923	3924	3925	3926	3927
7530	3928	3929	3930	3931	3932	3933	3934	3935
7540	3936	3937	3938	3939	3940	3941	3942	3943
7550	3944	3945	3946	3947	3948	3949	3950	3951
7560	3952	3953	3954	3955	3956	3957	3958	3959
7570	3960	3961	3962	3963	3964	3965	3966	3967
7600	3968	3969	3970	3971	3972	3973	3974	3975
7610	3976	3977	3978	3979	3980	3981	3982	3983
7620	3984	3985	3986	3987	3988	3989	3990	3991
7630	3992	3993	3994	3995	3996	3997	3998	3999
7640	4000	4001	4002	4003	4004	4005	4006	4007
7650	4008	4009	4010	4011	4012	4013	4014	4015
7660	4016	4017	4018	4019	4020	4021	4022	4023
7670	4024	4025	4026	4027	4028	4029	4030	4031
7700	4032	4033	4034	4035	4036	4037	4038	4039
7710	4040	4041	4042	4043	4044	4045	4046	4047
7720	4048	4049	4050	4051	4052	4053	4054	4055
7730	4056	4057	4058	4059	4060	4061	4062	4063
7740	4064	4065	4066	4067	4068	4069	4070	4071
7750	4072	4073	4074	4075	4076	4077	4078	4079
7760	4080	4081	4082	4083	4084	4085	4086	4087
7770	4088	4089	4090	4091	4092	4093	4094	4095

APPENDIX C

NUMERIC INPUT-OUTPUT CODES

a. The following codes are employed in numeric input-output in the BCD and Octal modes:

Teletype Meaning		FADAC Meaning			Fieldata Meaning	
Character	Tape Code	Octal Mode	BCD Mode	Binary Conversion	Tape Code	Character
0, P	01101	0	0	0000	00110000	0
1, Q	11101	1	1	0001	10110001	1
2, W	11001	2	2	0010	10110010	2
3, E	10000	3	3	0011	00110011	3
4, R	01010	4	4	0100	10110100	4
5, T	00001	5	5	0101	00110101	5
6, Y	10101	6	6	0110	00110110	6
7, U	11100	7	7	0111	10110111	7
8, I	01100	Halt	8	1000	10111000	8
9, O	00011	Compute	9	1001	00111001	9
(+) ", Z	10001	Fill	+	1010	00100010	+
-, A	11000	Verify	-	1011	00100001	-
., M	00111	Location	.	1100	10111101	.
/, X	10111	Clear	Clear	1101	11011101	X
Car. Ret	00010	Enter	Enter	1110	01000100	Car. Ret
Blank	00000	Blank	Blank	1111	00000000	Blank

b. The line through the tape code field indicates the position of the sprocket holes. Code is not applicable to magnetic tape.

APPENDIX D

ALPHANUMERIC INPUT-OUTPUT CODES

a. The following paper tape codes are used in alphanumeric input-output:

b. The line through the tape code field indicates the position of the sprocket holes.

c. The bit positions of the above codes are numbered 1 through 8 from right to left. Bit number 8 is parity bit; bit number 7 is control bit; bits number 6 through 1 are information bits.

FIELDATA Tape Code	FIELDATA Upper Case	Lower	FADAC Binary Code	FIELDATA Tape Code	FIELDATA Upper Case	Lower	FADAC Binary Code
11000 000	Master Sp	∿∿	000000	10100 000	)		100000
01000 001	Upper case	∿	000001	00100 001	–		100001
01000 010	Lower case	%	000010	00100 010	+		100010
11000 011	Tab	◇	000011	10100 011	<		100011
01000 100	Car. ret.	<	000100	00100 100	=		100100
11000 101	Space	△	000101	10100 101	>		100101
11000 110	A	A	000110	10100 110			100110
01000 111	B	B	000111	00100 111	$		100111
01001 000	C	C	001000	00101 000	*		101000
11001 001	D	D	001001	10101 001	(		101001
11001 010	E	E	001010	10101 010	"		101010
01001 011	F	F	001011	00101 011	:		101011
11001 100	G	G	001100	10101 100	?		101100
01001 101	H	H	001101	00101 101	!		101101
01001 110	I	I	001110	00101 110	,		101110
11001 111	J	J	001111	10101 111	Stop ⊕		101111
01010 000	K	K	010000	00110 000		0	110000
11010 001	L	L	010001	10110 001		1	110001
11010 010	M	M	010010	10110 010		2	110010
01010 011	N	N	010011	00110 011		3	110011
11010 100	O	O	010100	10110 100		4	110100
01010 101	P	P	010101	00110 101		5	110101
01010 110	Q	Q	010110	00110 110		6	110110
11010 111	R	R	010111	10110 111		7	110111
11011 000	S	S	011000	10111 000		8	111000
01011 001	T	T	011001	00111 001		9	111001
01011 010	U	U	011010	00111 010		'	111010
11011 011	V	V	011011	10111 011		;	111011
01011 100	W	W	011100	00111 100		/	111100
11011 101	X	X	011101	10111 101			111101
11011 110	Y	Y	011110	10111 110	Special □		111110
01011 111	Z	Z	011111	00111 111	Back space		111111

APPENDIX E

OPTIMIZATION TABLE

Operation Code	Execution Time	Optimum Operand Sector*	Optimum Next I Sector
CLA	1	t + 1	t + 1
CLS	1	t + 1	t + 1
ADD	1	t + 1	t + 1
SUB	1	t + 1	t + 1
MPL	18	t + 1	t+1 t+18
DIV	18	t + 1	t+1 t+18
STA[1]	1	t − 1 *	t + 1
STN[1]	1	t − 1 *	t + 1
STD[1]	1	t − 1 *	t + 1
STL[1]	1	t − 1 *	t + 1
STO	1-R&Q; 3-anything else	t + 1	t+1 R&Q
STP	1-R&Q; 3-anything else	t + 1	t+3 **
LDR	16	t + 1	t+1 t+16
LDQ	16	t + 1	t+1 t+16
STR	18	t + 1	t+1 t+18
TZE	1	t + 1	t + 1
TPL	1	t + 1	t + 1
TRA	1	t + 1	t + 1
TOV	1	t + 1	t + 1
ARC LLS	Variable	Variable	Variable
EXT	1	t + 1	t + 1
ABS	1	N/A	t + 1
RML	1	N/A	t + 1

* t − 1 in main memory; t + 1 in R or Q-Loops.

** t + 3 in main memory.

OPTIMIZATION TABLE (Cont'd)

Operation Code	Execution Time	Optimum Operand Sector	Optimum Next I Sector
RMN	1	N/A	t + 1
ZEL	1	N/A	t + 1
EQS	Variable	Variable	Variable
GES		Variable	Variable
HLT	1	N/A	t + 1
HDM	1	N/A	t + 1
IDM	1	N/A	t + 1
DIA	1	N/A	t + 1
DOF NSL	1	N/A	t + 1

APPENDIX F

SUMMARY OF FADAC INSTRUCTION REPERTOIRE

(Note: M is the Operand Address
[M] indicates Contents of M)

Operation	Mnemonic	Code	Effect of Flag & Notes	No. of Word Times for Execution
I. ARITHMETIC				
Add	ADD	00	ADD* = [M] → N	1
Subtract	SUB	02	SUB* = [M] → N	1
Multiply	MPY	20	No Flag. [M] → N	18
Clear & Add	CLA	24	CLA* = previous contents of A → L	1
Clear & Subtract	CLS	26	CLS* = previous contents of A → L	1
Divide	DIV	30	DIV* = unrounded Quotient in A. Remainder in L DIV = rounded Quotient in A Contents of L meaningless For both DIV & DIV* [M] → N	18
II. STORE & LOAD				
Store N	STN	40	No flag	1

Store D	STD	42	STD* = D.i → N
			D.i = D.0 if M is even;
			D.i = D.1 if M is odd
				1

Store A	STA	50	STA* = A → N	1
Store L	STL	52	STL* = L → N	1
Store Op Ad	STO	70	No flag	1 if M in R or Q 3 if M in Main Memory
Store (N1) Prog Ad	STP	60	No flag	1 if M in R or Q 3 if M in Main Memory
Store R loop	STR	62	No flag (Note: No store Q instruction exists)	18
Load R loop	LDR*	72	LDR* = R loop filled LDQ (no flag) = Q loop filled Note: flag determines whether R or Q loop is filled. Code 72 is used for both.	16
Load Q loop	LDQ	72	LDR* = R loop filled LDQ (no flag) = Q loop filled Note: flag determines whether R or Q loop is filled. Code 72 is used for both.	16
III. TRANSFER				
Transfer on Plus	TPL	10	TPL* = Transfer instruction → N	1
Transfer on Zero	TZE	12	TZE* = Transfer instruction → N	1
Unconditional Transfer	TRA	14	TRA* = Transfer instruction → N	1
Transfer on Overflow	TOV	16	TOV* = Transfer instruction → N	1

Operation	Mnemonic	Code	Effect of Flag & Notes	No. of Word Times for Execution
IV. SHIFT & CYCLE				
A right cycle	ARC	7600	NOTE: Place number of bits to be shifted or cycled in sector portion of Op Ad. Channel portion is part of Op code, as shown. Flag in all shift & cycle instructions indicates computer should test for overflow. (See Manual)	Let S = # of bits shifted. Let K = 0 when S is odd. Let K = 1 when S is even. Ex Time = (K/2) + (1/2(S+1)) (expressed as an Integer)
A right shift	ARS	7602		
A left cycle	ALC	7604		
A left shift	ALS	7606		
Long right cycle	LRC	7620		
Long right shift	LRS	7622		
Long left cycle	LLC	7624		
Long left shift	LLS	7626		
V. SPECIAL				
Extract	EXT	34	No flag	1
Equal Search	EQS	64	EQS* or GES* — If agreement reached, previous contents of A → L. If no agreement, L is unchanged.	Variable
Greater than or Equal Search	GES	66		
Halt	HLT	3720	No flag	1
Halt Compute Mode	HCM	3724	No flag	1
Initiate Compute Mode	ICM	3726	No flag	1
Zero L	ZEL	3762	No flag	1
Initiate Display Mode	IDM	3764	No flag	1
Halt Display Mode	HDM	3766	No flag	1
Take absolute Value	ABS	3770	ABS* = If A is neg, neg value → L; If A is pos, nothing to L	1 / 1
Replace A on Minus from L	RML	3772	When A is positive: RML = RML* = No change in A or L. When A is negative: RML = Complement of L → A; RML* = Complement of L → A, Negative contents originally in A → L.	1
Replace A on Minus from N	RMN	3774	When A is positive: RMN = RMN* = No change in A, N or L. When A is negative: RMN = Complement of N → A, N is unchanged, L is unchanged; RMN* Complement of N → A, N is unchanged, Negative contents originally in A → L.	

Note: Bits which normally designate the channel of the Op Ad are part of the Op Code of most Commands in Sections IV, V, and VI.

Operation	Mnemonic	Code	Effect of Flag & Notes	No. of Word Times for Execution
VI. SERIAL I/O				
Discrete Input to A	DIA	3640	No flag	1
Discrete Outp Off	DOF	3700	No flag	1
Outp Dev Stepping - 1	OD1	3702	No flag	1
Outp Dev Stepping - 2	OD2	3704	No flag	1
Outp Dev Stepping - 3	OD3	3706	No flag	1
Inp Dev Stepping - 1	ID1	3710	No flag	1
Inp Dev Stepping -2	ID2	3712	No flag	1
Inp Dev Stepping -3	ID3	3714	No flag	1
No Solution Light	NSL	3716	No flag	1

VII. PARALLEL INPUT/OUTPUT COMMAND SUMMARY

In all Parallel I/O commands, a flag indicates input (output) by WORD, no flag indicates input (output) by CHARACTER. No Op Ad is allowed in I/O commands. The bits which normally designate the channel of the Op Ad are part of the I/O Op code. Bits which normally designate the sector of the Op Ad are used to designate the number of characters (words) to be input (output), less one. Execution time for all I/O is variable.

Mnemonic	Op Code	Interpretation	Comments
WEOT	4400	Write FADAC to Ext Dev in Oct (TT)	
WFOT	4420	Write FADAC to FADAC in Oct (TT)	
WEOF	4440	Write FADAC to Ext Dev in Oct (FD)	
WFOF	4460	Write FADAC to FADAC in Oct (FD)	
WEDT	4500	Write FADAC to Ext Dev Dec (TT)	
WFDT	4520	Write FADAC to FADAC Dec (TT)	
WEDF	4540	Write FADAC to Ext Dev Dec (FD)	
WFDF	4560	Write FADAC to FADAC Dec (FD)	
WE6	4600	Write FADAC to Ext Dev α-6	
WF6	4620	Write FADAC to FADAC α-6	
WE5*	4700	Write FADAC to Ext Dev α-5	Output is by word only. Flag required.
WF5*	4720	Write FADAC to FADAC α-5	Output is by word only. Flag required.
WE4	4700	Write FADAC to Ext Dev α-4	Output is by character only. Flag must be omitted.
WF4	4720	Write FADAC to FADAC α-4	Output is by character only. Flag must be omitted.
REO	5400	Rd Ext Dev in Oct	
RTO	5420	Rd Tape Dev in Oct	
RKO	5460	Rd Keyboard in Oct	
RED	5500	Rd Ext Dev in Dec	
RTD	5520	Rd Tape Dev in Dec	
RKD	5560	Rd Keyboard in Dec	
RE6	5600	Rd Ext Dev in α-6	
RT6	5620	Rd Tape Dev in α-6	
RM6	5640	Rd Mag Tape in α-6	
RK6	5660	Rd Keyboard in α-6	
RE5	5700	Rd Ext Dev in α-5	
RT5	5720	Rd Tape Dev in α-5	
RM5	5740	Rd Mag Tape in α-5	
RK5	5760	Rd Keyboard in α-5	

SUMMARY OF CONTROL BUTTON ADDRESSES

Control Button	Address
Sample Matrix	000.00
Test	000.01
Set Up	000.02
Recall	000.03
Send	000.04
Compute	000.05
Trig	000.06
Receive	000.07

SUMMARY OF REGISTER ADDRESSES

Register	Address
R	142. (00 to 17)
Q	152. (00 to 17)
D.0	160.00
D.1	160.01
A	170.00
L	172.00
N	174.00

TIMING: { 1 word time = 78 micro-sec
1 drum revolution = 10 milli-sec

MAXIMUM I/O RATES: { 4000 characters/sec in Alpha 6
3550 characters/sec in Alpha 5
650 characters/sec in Oct or Dec

SUMMARY OF MEMORY DESIGNATIONS

Memory I	-	Channels 00-76
Memory II	-	Channels 100-136
Memory III	-	Channels 300-336

WORKING STORAGE DESIGNATIONS

Size of WS	WS-Channels
4	70-76
12	70-76 110-116 130-136
16	40-76