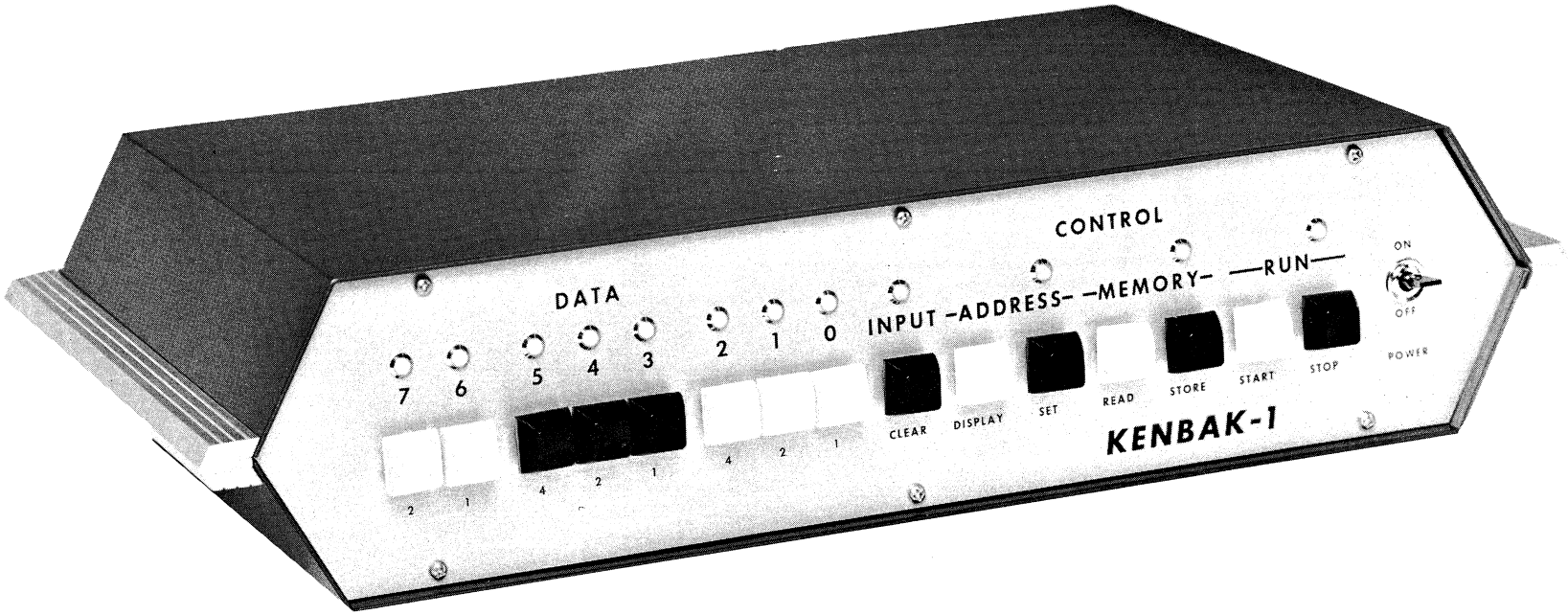
Programming Reference Manual

KENBAK-1 Computer

KENBAK
P. O. BOX 49324
LOS ANGELES, CA. 90049



frontispiece

4/1/71

Preface

This manual is a summary of the KENBAK-1 computer programming characteristics. The reader is assumed to be acquainted with the general theory of digital computers. Other manuals published by KENBAK have as their subjects the general theory of computers, programming of the KENBAK-1, and the maintenance and theory of operation of the KENBAK-1.

To correct errors or to clarify the presentation, revised pages will be issued from time to time. The list of current pages should be retained as it indicates the status of the manual. Pages are punched for a three ring binder so that the user may keep all of his material in one place with easy accessibility.

Questions and comments are welcomed since that provides the feedback which tells how well the job is being done.

Price: $2.00

Programming Reference Manual
KENBAK-1 Computer

List of Current Pages, Revision 0

Page	Dated	Page	Dated	Page	Dated
PR-1	4-1-71	PR-9	4-1-71	PR-17	4-1-71
PR-2	4-1-71	PR-10	4-1-71	PR-18	4-1-71
PR-3	4-1-71	PR-11	4-1-71	PR-19	4-1-71
PR-4	4-1-71	PR-12	4-1-71	PR-20	4-1-71
PR-5	4-1-71	PR-13	4-1-71	PR-21	4-1-71
PR-6	4-1-71	PR-14	4-1-71	PR-22	4-1-71
PR-7	4-1-71	PR-15	4-1-71	PR-23	4-1-71
PR-8	4-1-71	PR-16	4-1-71	PR-24	4-1-71

Errata

For which no revised page was issued

NONE

Programming Reference Manual
KENBAK-1 Computer

Table of Contents

INTRODUCTION

The KENBAK-1 is a serial stored program general purpose digital computer with a memory of 256 eight bit bytes. Internally the machine is binary two's complement. Halt, No-op, and Shift instructions require one byte. All other instructions require two bytes which may start on either an odd or even address.

MEMORY STRUCTURE AND ADDRESSING

Each of the 256 bytes of memory is addressable. The octal range of their addresses is 000 to 377 (decimal 000 to 255). Since 2^8 is equal to 256, one byte of memory can hold a complete address. The eight bits of a byte are always identified as:

b7 b6 b5 b4 b3 b2 b1 b0	Binary to Octal Conversion
x x	Octal group 100
x x x	Octal group 10
x x x	Octal group 1

where the most significant bit is b7 and the least significant bit is b0. Octal group 100 never assumes a value larger than 3.

SPECIAL MEMORY LOCATIONS

Nine memory locations are used for special purposes. They may be addressed and used as an operand location though the programmer must be aware that he may be interfering with the assigned special function. The special locations are:

Octal Location	Purpose
000	A Register
001	B Register
002	X Register
003	P Register
200	Output Register
201	Overflow and Carry for the A Register
202	Overflow and Carry for the B Register
203	Overflow and Carry for the X Register
377	Input Register

The A Register is the primary register of the arithmetic unit. Operations that can be performed with it include Load, Store, Add, Subtract, And, Or, Load Complement, Shifts and Rotates. The B Register is the secondary register of the arithmetic unit. Operations that can be performed with it include Load, Store, Add, Subtract, Shifts, and Rotates. The contents of the X Register are used in the Indexed address mode. Operations that can be performed with it include Load, Store, Add, Subtract. When not being used in Indexed addressing, the X Register can be used as an arithmetic register.

The A, B, and X Registers can all be used (singly, not jointly) to control the branch conditions. The testable conditions include Non-zero, Zero, Less than Zero, Positive (including zero), and Positive Non-zero.

The P Register is the Program Counter which shows the address of the next instruction to be executed.

The Output Register, location 200, controls the Console Data Lamps when the computer is in the Run state.

Locations 201, 202, and 203 contain the Overflow (OF) and Carry (CA) bits for the A, B, and X Registers, respectively.

b7	b6	b5	b4	b3	b2	b1	b0
0	0	0	0	0	0	CA	OF

Each time that an Add or Subtract instruction is performed, the Overflow and Carry bits for the register are updated. They remain in this state until the next Add or Subtract operation to that register occurs. The remaining six bits in each word are always set to zero when the Overflow and Carry are updated. The Load Complement and the Shift instructions do not alter these words, nor do any other instructions.

Location 377 is the input location. Data from the Console Data Pushbuttons is placed here in both the Run and Halt states of the computer.

NUMBER REPRESENTATIONS

Several choices exist for number representation including positive integer, signed integer, and signed fractional. The range of numbers that can be expressed in one byte in each mode is shown in the following table:

Number Representation	Decimal Value	Octal Code	Binary Code
Positive Integer			
Largest positive number	+255	377	11 111 111.
Smallest number	0	000	00 000 000.
Signed Integer			
Largest positive number	+127	177	01 111 111.
Smallest negative number	-128	200	10 000 000.
Signed Fractional			
Largest positive number	+127/128	177	0.1 111 111
Smallest negative number	-128/128	200	1.0 000 000

The Jump (Branch) conditions of Positive, Negative, and Positive Non-zero assume one of the signed modes of number representation. The Jump conditions of Zero and Non-zero apply for all of the representations since 00 000 000 is the binary code for zero in all of them.

As detected by the computer, Overflow is based on the signed representations. Since the binary arithmetic is identical in all of these representations, the Carry bit is not affected by the programming choice for number representation. For positive integer representation the Carry is also the Overflow condition.

The range of quantities can be extended by using more than one byte.

ADDRESSING MODES

The Add, Subtract, Load, Store, Load Complement, And, and Or instructions have five addressing modes:

1. Constant, or Immediate
2. Memory
3. Indirect, or Deferred
4. Indexed
5. Indirect Indexed

In the Constant, or Immediate, mode of addressing, the operand is the contents of the second word of the instruction.

In the Memory mode of addressing, the second word of the instruction is the address of the operand.

In the Indirect addressing mode, the second word of the instruction is the address of the address of the operand.

In the Indexed mode of addressing, the contents of the second word of the instruction are added to the contents of the X Register to form the address of the operand.

In the Indirect Indexed mode of addressing, the contents of the second word of the instruction are used as an address pointer to the contents of another, second, location. The contents of this second location are added to the contents of the X Register to form the address of the operand.

In the Store instruction the operand address is where the data is stored. In Store Immediate instructions the data is stored in the second half of the instruction.

The Bit Manipulation instructions have only the Memory addressing mode.

The Jump or Branch instructions have Memory (called Direct) addressing or Indirect addressing.

INSTRUCTIONS

In the following pages, the individual instructions are discussed. It is assumed in each case that the instruction is in location p and (if it is a two byte instruction) in p + 1.

ADDITION (ADD)

First Byte										Second Byte
Register	7	6	5	4	3	2	1	0	Addressing Mode	7 6 5 4 3 2 1 0
A	0	0	0	0	0	0	1	1	Immediate	
B	0	1				1	0	0	Memory	Address or
X	1	0				1	0	1	Indirect	Operand
						1	1	0	Indexed	
						1	1	1	Indirect, Indexed	

The operand is specified or located according to the rules for the addressing mode. The operand and the contents of the designated register are added together and the sum is placed in the designated register. The operand is unchanged in the memory (unless the operand was in the designated register itself). The Carry and Overflow bits are updated in location 201 (for A), in 202 (for B), or in 203 (for X). The P Register is advanced by 2 and the next instruction is taken from p + 2.

SUBTRACTION (SUB)

First Byte										Second Byte
Register	7	6	5	4	3	2	1	0	Addressing Mode	7 6 5 4 3 2 1 0
A	0	0	0	0	1	0	1	1	Immediate	
B	0	1				1	0	0	Memory	Address or
X	1	0				1	0	1	Indirect	Operand
						1	1	0	Indexed	
						1	1	1	Indirect, Indexed	

The operand is specified or located according to the rules for the addressing mode. The operand is subtracted from the contents of the designated register and the difference is placed in the designated register. The operand is unchanged in the memory (unless the operand was in the designated register itself). The Carry (or Borrow) and the Overflow bits are updated in location 201 (for A), in 202 (for B), or in 203 (for X). The P Register is advanced by 2 and the next instruction is taken from p + 2.

LOAD (LOAD)

First Byte										Second Byte
Register	7	6	5	4	3	2	1	0	Addressing Mode	7 6 5 4 3 2 1 0
A	0	0	0	1	0	0	1	1	Immediate	
B	0	1				1	0	0	Memory	Address or
X	1	0				1	0	1	Indirect	Operand
						1	1	0	Indexed	
						1	1	1	Indirect, Indexed	

The operand is specified or located according to the rules for the addressing mode. The operand is placed in the designated register replacing the previous contents. The operand is unchanged in the memory. The P Register is advanced by 2 and the next instruction is taken from p + 2.

STORE (STORE)

First Byte										Second Byte
Register	7	6	5	4	3	2	1	0	Addressing Mode	7 6 5 4 3 2 1 0
A	0	0	0	1	1	0	1	1	Immediate	
B	0	1				1	0	0	Memory	Address or
X	1	0				1	0	1	Indirect	Operand
						1	1	0	Indexed	
						1	1	1	Indirect, Indexed	

The location to receive the data is determined by the addressing rules. The contents of the designated register are placed in this location replacing the previous contents. The contents of the register are unchanged. The P Register is advanced by 2 and the next instruction is taken from p + 2.

LOGICAL PRODUCT (AND)

First Byte									Second Byte
7	6	5	4	3	2	1	0	Addressing Mode	7 6 5 4 3 2 1 0
1	1	0	1	0	0	1	1	Immediate	
				1	0	0		Memory	Address or
				1	0	1		Indirect	Operand
				1	1	0		Indexed	
				1	1	1		Indirect, Indexed	

The operand is specified or located according to the rules for the addressing mode. Each bit of the operand is Anded with the corresponding bit of the A Register, according to the truth table below, and the result is placed in the A Register replacing the previous contents. The operand is unchanged in the memory.

The P Register is advanced by 2 and the next instruction is taken from p + 2.

Bits from Two Operands		Result	
		AND	(Inclusive) OR
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

LOGICAL SUM (OR)

First Byte									Second Byte
7	6	5	4	3	2	1	0	Addressing Mode	7 6 5 4 3 2 1 0
1	1	0	0	0	0	1	1	Immediate	
				1	0	0		Memory	Address or
				1	0	1		Indirect	Operand
				1	1	0		Indexed	
				1	1	1		Indirect, Indexed	

The operand is specified or located according to the rules for the addressing mode. Each bit of the operand is Or-ed with the corresponding bit of the A Register, according to the truth table above (see AND), and the result is placed in the A Register replacing the previous contents. The operand is unchanged in the memory.

The P Register is advanced by 2 and the next instruction is taken from p + 2.

LOAD COMPLEMENT (LNEG)

First Byte									Second Byte
7	6	5	4	3	2	1	0	Addressing Mode	7 6 5 4 3 2 1 0
1	1	0	1	1	0	1	1	Immediate	
					1	0	0	Memory	Address or
					1	0	1	Indirect	Operand
					1	1	0	Indexed	
					1	1	1	Indirect, Indexed	

The operand is specified or located according to the rules for the addressing mode. The A Register is loaded with the arithmetic complement (the value obtained by subtracting from zero) of the operand. This replaces the previous contents of the A Register. The operand is unchanged in the memory unless it had been specified as the contents of the A Register itself.

Overflow occurs if the operand was equal to -128 (10 00 000). In this case the result obtained is -128. This overflow is not detected nor is the A Register Overflow bit in location 201 altered by the LNEG instruction.

The P Register is advanced by 2 and the next instruction is taken from p + 2.

JUMP DIRECT (JPD)
JUMP INDIRECT (JPI)
JUMP AND MARK DIRECT (JMD)
JUMP AND MARK INDIRECT (JMI)

First Byte										Second Byte
	7	6	5	4	3	2	1	0	Condition	7 6 5 4 3 2 1 0
A Register	0	0	1			0	1	1	Non-zero	
B Register	0	1				1	0	0	Zero	Target Address or
X Register	1	0				1	0	1	Negative	Indirect Reference to
Unconditional	1	1				1	1	0	Positive	Target Address
						1	1	1	Positive Non-zero	
Jump				0	0				Direct Address (Memory)	
Jump and Mark				1	1				Indirect Address	

The "target" address is defined to be:

Jump Direct or Jump and Mark Direct	The contents of the second byte of the instruction
Jump Indirect or Jump and Mark Indirect	The address contained in the byte whose address is in the second byte of the instruction

The Direct address mode is the same as the Memory address mode. The Indirect address mode for the Jump instructions is the same as the Indirect mode for the arithmetic and logic instructions.

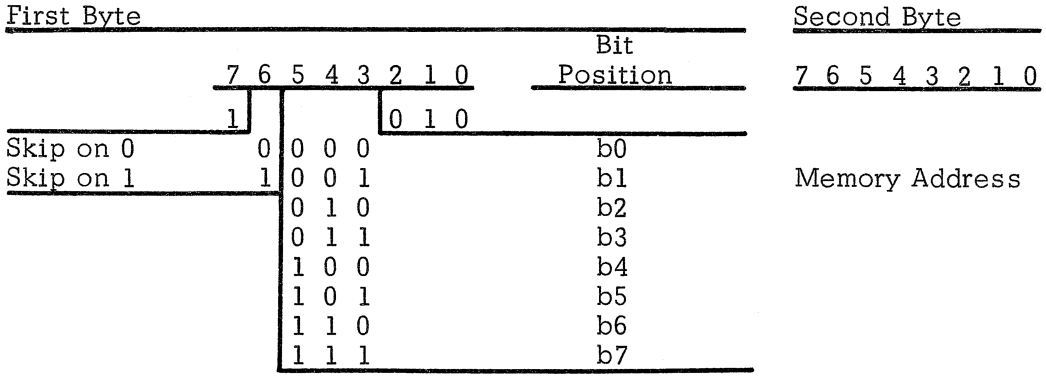
The designated register (A, B, or X) is examined to determine if the Jump Conditions are met (true). A Non-zero condition in the register means at least one bit is a One. A Zero condition in the register means that all bits in the register are Zero. The Negative condition means that b7 (the most significant bit) is a One. The Positive condition means that b7 is a Zero. The Positive Non-zero condition means b7 is a Zero and at least one of the remaining seven bits is a One. The Jump Conditions can be forced to be true by the Unconditional specification.

If the Jump Conditions are true because the specified condition is true or because an Unconditional Jump is specified, then the next instruction is not taken from location p + 2, but from the target address in the Jump instructions or from the target address plus one in the Jump and Mark instructions. If the Jump Conditions are not true, the next instruction is taken from p + 2.

In the Jump and Mark instructions, the value p + 2 is stored in the target address prior to executing the instruction in the location following the target address.

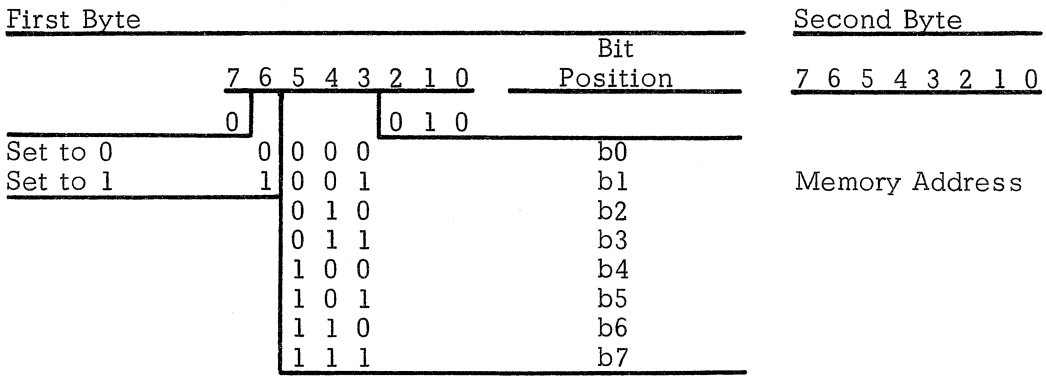
Notice that the instruction will not be a Jump instruction of any type unless bits 2, 1, and 0 of the first byte are either 011, 100, 101, 110, or 111 even if the Jump Conditions are Unconditional.

SKIP ON ZERO (SKP 0)
SKIP ON ONE (SKP 1)



First the location in memory whose address is given in the second byte of the instruction is determined. Within these eight bits, the one is selected whose position corresponds to the three bit code in positions 5, 4, and 3 of the first byte of the instruction. If the instruction is Skip on Zero and the selected bit is a Zero or if the instruction is Skip on One and the selected bit is a One, then the next instruction is taken from p + 4 and the next two locations in the "normal" instruction sequence are not used. Either a two byte instruction or two one byte instructions may be skipped. If the Skip Conditions are not met, then the next instruction is taken from p + 2.

SET 0 (SET 0)
SET 1 (SET 1)



The SET 0 instruction forces a single bit in the memory to a Zero. The SET 1 instruction forces a single bit in the memory to a One. The address of the memory location is contained in the second byte of the instruction. The bit within this word is determined by the bit code in positions 5, 4, and 3 of the first byte of the instruction. No other bit in the memory is changed except the designated one. The P Register is advanced by 2 and the next instruction is taken from location p + 2.

SHIFT LEFT (SFTL)
SHIFT RIGHT (SFTR)
ROTATE LEFT (ROTL)
ROTATE RIGHT (ROTR)

First Byte — One byte instruction

	7	6	5	4	3	2	1	0	
						0	0	1	
Right	0								
Left	1			0	1				1 Place
Shift		0		1	0				2 Places
Rotate		1		1	1				3 Places
A Register			0	0	0				4 Places
B Register			1						

Shifts and Rotates may be for 1, 2, 3, or 4 places. An n place Shift or Rotate is equal to performing a 1 place Shift or Rotate n times. The 1 place Shifts and Rotates are defined below.

		b7	b6	b5	b4	b3	b2	b1	b0
SHIFT LEFT 1	Original	c7	c6	c5	c4	c3	c2	c1	c0
	After	c6	c5	c4	c3	c2	c1	c0	0
ROTATE LEFT 1	Original	c7	c6	c5	c4	c3	c2	c1	c0
	After	c6	c5	c4	c3	c2	c1	c0	c7
SHIFT RIGHT 1	Original	c7	c6	c5	c4	c3	c2	c1	c0
	After	c7	c7	c6	c5	c4	c3	c2	c1
ROTATE RIGHT 1	Original	c7	c6	c5	c4	c3	c2	c1	c0
	After	c0	c7	c6	c5	c4	c3	c2	c1

This Shift or Rotate is made to the A or B Register as specified. The P Register advances by one and the next instruction is taken from p + 1.

NO OPERATION (NOOP) One byte instruction

First Byte

7	6	5	4	3	2	1	0
1	x	x	x	x	0	0	0

The P Register is advanced by one and the next instruction is taken from location p + 1.

HALT (HALT) One byte instruction

First Byte

7	6	5	4	3	2	1	0
0	x	x	x	x	0	0	0

At the end of this instruction, the computer goes from the Run state to the Halt state. The P Register is advanced and ready to take the next instruction from location p + 1.

SYMBOLIC REPRESENTATION OF INSTRUCTIONS

An instruction, including the operand address or specification, is represented symbolically in three parts. The first field is the basic operation such as ADD or SFTR. Preferred abbreviations for these operations were given with the instruction descriptions. The second field is the details of the basic operation, for example, "A" meaning to add to the A Register or B3 meaning to shift the B Register 3 places. The third field is the operand address except in the case of the Constant addressing mode which has a special representation illustrated below. Indirect addressing is indicated by the use of parentheses around the address. The Index addressing mode is indicated by ",X". The third field for the five addressing modes becomes:

Constant	C = Value of Operand
Memory	DATE
Indirect	(DATE)
Indexed	DATE, X
Indirect/Indexed	(DATE), X

where DATE is the name of some location in memory. One byte instructions have no third field and may (Shifts and Rotates) or may not (Halt and Noop) have a second field.

REGISTER-TO-REGISTER OPERATIONS

All of these operations are produced by a single instruction using the Memory addressing mode. They are special cases of the previously discussed instructions which result from the addressability of the A, B, X, and P Registers.

Operation	Symbolic Representation
CLEAR A	SUB A A
CLEAR B	SUB B B
CLEAR X	SUB X X
TRANSFER A TO B	LOAD B A or STORE A B
TRANSFER A TO X	LOAD X A or STORE A X
TRANSFER B TO A	LOAD A B or STORE B A
TRANSFER B TO X	LOAD X B or STORE B X
TRANSFER X TO A	LOAD A X or STORE X A
TRANSFER X TO B	LOAD B X or STORE X B
LEFT SHIFT A 1	ADD A A
LEFT SHIFT B 1	ADD B B
LEFT SHIFT X 1	ADD X X

This left shift operation is the same as produced by the instruction of the same name. When produced by adding the register to itself, Overflow and Carry are detected which they are not by the Left Shift instruction. The Carry bit is set if b7 was a One and is reset if b7 was a Zero. The Overflow bit is set if the original number was larger than 63 or smaller than -64. Otherwise the Overflow bit is reset.

LOAD A NEG A	LNEG A A
LOAD A NEG B	LNEG A B
LOAD A NEG X	LNEG A X

The A Register is loaded with the complement (the value obtained by subtracting from 00 000 000) of the specified register. Overflow, undetected, occurs when the original number is -128 when the result produced is also -128.

A + B TO A	ADD A	B
A - B TO A	SUB A	B
A + X TO A	ADD A	X
A - X TO A	SUB A	X
B + A TO B	ADD B	A
B - A TO B	SUB B	A
B + X TO B	ADD B	X
B - X TO B	SUB B	X
X + A TO X	ADD X	A
X - A TO X	SUB X	A
X + B TO X	ADD X	B
X - B TO X	SUB X	B
A AND B TO A	AND A	B
A AND X TO A	AND A	X
A OR B TO A	OR A	B
A OR X TO A	OR A	X
JUMP TO A + 2	STORE A	P
JUMP TO B + 2	STORE B	P
JUMP TO X + 2	STORE X	P

Storing a number n in the P Register under program control causes an unconditional branch or jump to location n + 2.

TABLE LOOK UP

Using the contents of the A, B, or X Register as an Indirect address produces a result which has been called "table look up". The operation

LOAD A (A)

replaces the address in the A Register by the contents of that address. Similar operations can be produced by the B and X Registers or by combinations of the A, B, and X Registers.

RELATIVE OPERAND ADDRESSING

Using the Indirect Indexed addressing mode with the P Register as the Indirect address location creates a condition which could be called "relative operand addressing". The operand address will be a "distance" from the instruction itself which is given by the contents of the X Register.

INPUT AND OUTPUT

Input from the Console Data Pushbuttons to the program and output from the program to the Console Data Lamps can occur while the computer is in the Run state. All input appears in location 377_8 whose contents may be tested with the Skip instructions or whose contents may be transferred to the A, B, or X Registers. It must be remembered that data will appear in location 377_8 while the operator is in the process of forming his input. If the input data is not to be used until the formation or entry is completed, one of the bits should be used as an interlock signal.

The contents of location 200_8 are continuously displayed in the Console Data Lamps when the computer is in the Run state. A transfer of data to 200_8 is equivalent to the output of that data to the Console Data Lamps. If the computer goes from the Run to the Halt state, the contents of location 200_8 are displayed until the operator makes another choice.

CONSOLE OPERATIONS

The Console switches and lamps provide the means to control the computer, to load programs and data, and to receive output (frontispiece).

Power On-Off Switch

The Power Switch controls the line voltage to the computer.

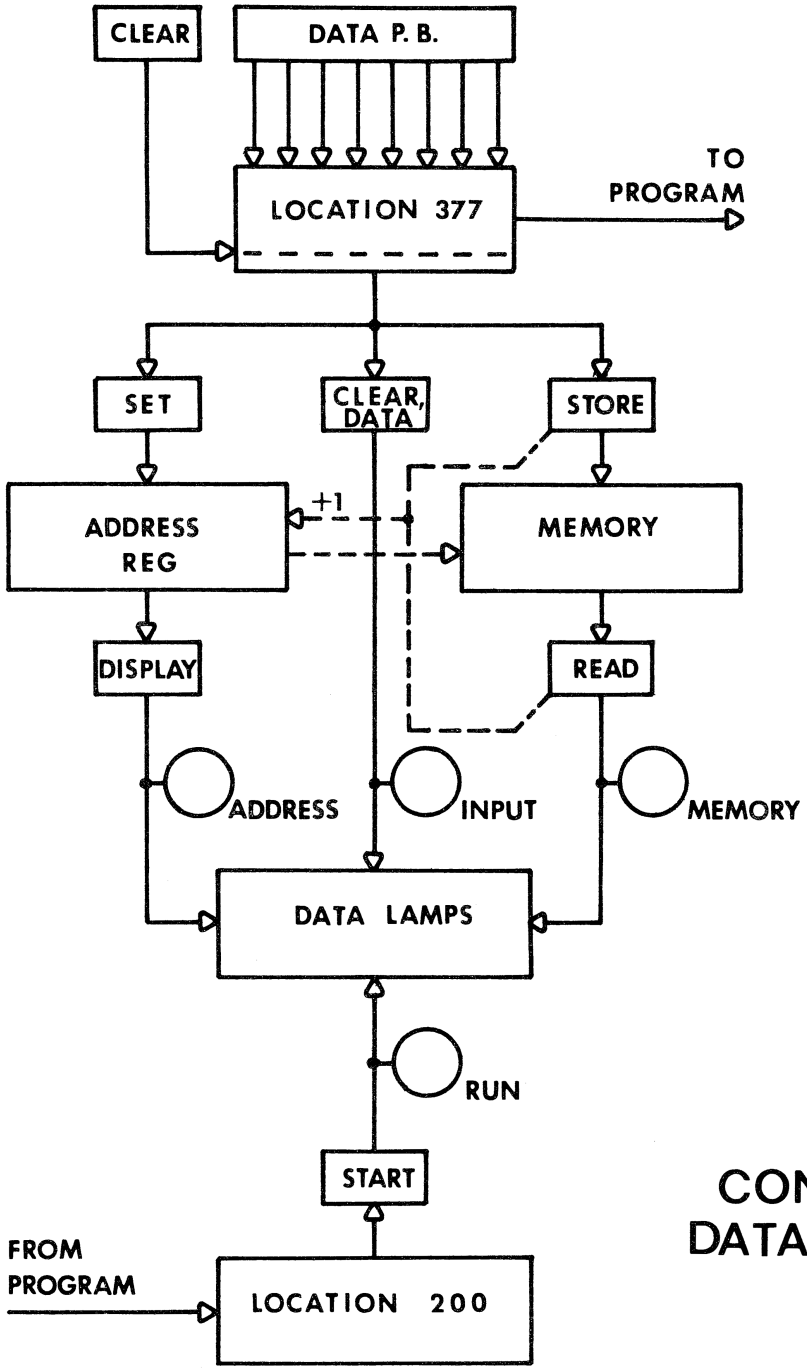
Data in the memory is lost when power is turned off. When power is turned on, random data will appear in the memory. Since the computer uses very little power (40 watts), it can be left on without damage or significant expense.

No indicator light per se is used to indicate the power status. If any of the data and control lamps are on, then power must be on. If power is thought to be on and none of the data or control lamps are on, pushing the Clear Pushbutton will cause the Input Lamp to turn on if power is available. Also the Start, Read, Display, or the eight Data Pushbuttons can be used to test the power condition. The fan can also be heard when power is on.

When power is turned on, the Start Pushbutton should be depressed. If the computer stays in the Run state, then the Stop Pushbutton should also be depressed. These operations will initialize the internal control circuits.

Console Data Lamps

The eight Console Data Lamps receive their information from one of four sources as indicated by the Console Control Lamps, page PR-19. If the Input Lamp is On, the Console Data Lamps are displaying the contents of location 377_8 which is the location in memory where input is assembled. If the Address Lamp is On, the Console Data Lamps show the contents of the Address Register which is used for reading or storing data in the memory. If the Memory Lamp is On, the Console Data Lamps show the contents of a memory location which has been read. If the Run Lamp is On, or if none of the Control Lamps are On, the Console Data Lamps show the contents of location 200_8 which is assigned as the program output register.



CONSOLE DATA FLOW

Console Data Lamps Show	When Control Lamp is On
Contents of 377_8	Input
Contents of 200_8	Run or None of the Control Lamps
Contents of Address Register	Address
Contents of Memory Location	Memory

Console Data Pushbuttons and Clear

The Clear Pushbutton erases the contents of location 377_8 to 00 000 000. A Data Pushbutton sets the bit position in 377_8 which corresponds to its respective number. If a One is entered by mistake, Clear must be used to erase location 377_8 to 00 000 000 and the correct Ones must be re-entered. Between the entry of different bytes of information, Clear is used to erase the previous data.

Entry can occur during the Run state. Normally, the Console Data Lamps are displaying the contents of location 200_8 during the Run state. During the actual interval while the Clear or Console Data Pushbuttons are depressed, the Console Data Lamps will predominantly display the contents of location 377_8. In the background, as a fainter light, the contents of location 200_8 will also be displayed. In the Run state, entry of data to location 377_8 must be accompanied by a stored program which will interpret and make use of the data.

In the Halt state, the data entered into 377_8 can be used to set the Address Register or it can be stored in the memory at locations determined by the Address Register.

Address Control Lamp, Address Display and Set Pushbuttons

While the computer is halted, data may be entered or read from the memory where the address to be used is held in the Address Register. Data is entered into the Address Register by the Address Set Pushbutton. The source of this data is location 377_8 which in turn is determined by the Clear and Console Data Pushbuttons. The contents of the Address Register may be caused to be displayed by the Address Display Pushbutton. This will be the next address. When reading or storing data in the memory, the contents of the Address Register are incremented by one for each depression of the Memory Read or Memory Store Pushbuttons.

When the computer goes into the Run state the value in the Address Register will be changed in an unspecified way.

Memory Control Lamp, Memory Read and Store Pushbuttons

While the computer is in the Halt state, data may be examined (read) or be entered (stored) in the memory. This may be done one byte at a time. The memory location to be used is contained in the Address Register. For storing data, the source of the data is location 377_8. In reading data, the Console Data Lamps will display the information. Each depression of the Read or Store Pushbutton will cause the Address Register to advance by one.

Start and Stop Pushbuttons

After a program and data have been loaded into the memory, the computer can be made to go into the Run state by the Start Pushbutton. In the Run state, the computer is executing a stored program. The Run state will last until a Halt instruction is encountered or until the Stop Pushbutton is depressed.

The computer can be made to execute instructions one at a time, stopping after each instruction is finished. To do this, hold the Stop Pushbutton depressed and push and release Start. The Stop Pushbutton can then be released. The computer will have executed one and only one instruction.

If the P Register is examined while the computer is halted, it will show the address of the next instruction to be executed. Similarly, if the P Register is being loaded, it should be set with the address of the next instruction to be executed.

Lamp Tests

To test whether a lamp is burned out push the Pushbutton in the table below. The lamp should light.

To Test	Push
Data Lamp 0	Data Pushbutton 0
Data Lamp 1	Data Pushbutton 1
Data Lamp 2	Data Pushbutton 2
Data Lamp 3	Data Pushbutton 3
Data Lamp 4	Data Pushbutton 4
Data Lamp 5	Data Pushbutton 5
Data Lamp 6	Data Pushbutton 6
Data Lamp 7	Data Pushbutton 7
Input	Clear
Address	Display (Address)
Memory	Read (Memory)
Run	Start

EXAMPLE OF CONSOLE USE

Load the program (all octal numbers)

	Location	Contents	Symbolic Instruction
	003	004	Initial value of P Reg.
LOOP	004	103	ADD B C=1
	005	001	
	006	134	STORE B OUTPUT
	007	200	
	010	344	JPD UNC LOOP
	011	004	

OUTPUT	200	xxx	

Steps:

1. Power On
2. Start
3. Stop
4. Clear
5. Enter 003
6. Set (Address)
7. Clear
8. Enter 004
9. Store (Memory)
10. Clear
11. Enter 103
12. Store
13. Clear
14. Enter 001
15. Store
16. Clear
17. Enter 134
18. Store
19. Clear
20. Enter 200
21. Store
22. Clear
23. Enter 344
24. Store
25. Clear
26. Enter 004
27. Store
28. Display (Address)

Step 27 concluded the entry process. Step 28 is a double check that the Address Register has the correct value which in this case should be 012, the next address. To check the entry of the data, which is a good habit to get into, or in general, to examine memory proceed with

29. Clear
30. Enter 003
31. Set (Address)
32. Read (should be 004)
33. Read (should be 103)
34. Read (should be 001)
35. Read (should be 134)
36. Read (should be 200)
37. Read (should be 344)
38. Read (should be 004)

To run the program

39. Start

This program adds 1 to the B Register, puts the value of B out to location 200_8, and hence the Data Lamps, and recycles or loops. The binary counting process should be in evidence on the Data Lamps though the lamps will be operating too fast in the lower order positions to observe the action in detail.

SUMMARY OF INSTRUCTION CODING

Instruction	First Byte Octal Digits		
	D--	-D-	--D
Add, Sub, Load, Store	A Reg = 0 B Reg = 1 X Reg = 2	Add = 0 Sub = 1 Load = 2 Store = 3	Constant = 3 Memory = 4 Indirect = 5 Indexed = 6 Ind/Ind = 7
Or, And, Lneg	3	Or = 0 (Noop = 1) And = 2 Lneg = 3	Constant = 3 Memory = 4 Indirect = 5 Indexed = 6 Ind/Ind = 7
Jumps	A Reg = 0 B Reg = 1 X Reg = 2 Unc. = 3	JPD = 4 JPI = 5 JMD = 6 JMI = 7	($\neq$ 0) = 3 (= 0) = 4 (< 0) = 5 ($\geq$ 0) = 6 (> 0) = 7
Bit Test and Manipulation	Set to 0 = 0 Set to 1 = 1 Skip on 0 = 2 Skip on 1 = 3	Digit Value = Position	2
Shifts, Rotates (one byte only)	Right Shift = 0 Right Rotates = 1 Left Shift = 2 Left Rotate = 3	A = 0 B = 4 ----plus---- 1 place = 1 2 places = 2 3 places = 3 4 places = 0	1
Miscellaneous (one byte only)	Halt = 0 or 1 Noop = 2 or 3	Any value	0