Systems Engineering Laboratories

SEL 810A General Purpose Computer Operating Instructions

Systems Engineering Laboratories

Operating Instructions

SEL 810A
General Purpose Computer

March 8, 1967

SEL 95064A
Price: $10.00

6901 West Sunrise Blvd., Fort Lauderdale, Florida 333
Area Code 305 587-29C

PREFACE

This manual provides basic information for controlling the operations of any SEL 810A Computer System. In addition to studying this manual, it is suggested that you become familiar with the other SEL 810A publications:

General Reference Manual

Interface Design Manual

Assembler Reference Manual

Loader Manual

Fortran IV Reference Manual

Hardware Diagnostics

Library Programs

TABLE OF CONTENTS

Section			Page
1	BASIC 810A SYSTEM		1-1
	1.1	Introduction	1-1
	1.2	Basic System	1-1
	1.2.1	Specifications	1-2
	1.2.2	Options	1-3
	1.2.3	Standard Software	1-3
	1.3	Installation Requirements	1-4
2	CONSOLE OPERATIONS		2-1
	2.1	Introduction	2-1
	2.2	Console Input/Output	2-1
	2.2.1	ASR-33 Teletype Unit (Standard)	2-1
	2.2.2	Keyboard Printer	2-1
	2.3	Printer Operation	2-1
	2.3.1	Local Operation	2-2
	2.3.2	On-Line Operation	2-2
	2.3.3	Keyboard Interlock	2-2
	2.4	Paper Tape Reader	2-2
	2.5	Paper Tape Punch	2-2
	2.6	Console Operations	2-2
	2.6.1	Example Console Operations	2-4
3	BASIC OPERATING PROCEDURES		3-1
	3.1	Console Control Panel Operations	3-1
	3.1.1	Normal Computer Turn-On	3-1
	3.1.2	Normal Computer Turn-Off	3-2
	3.1.3	Mode Selection	3-2
	3.2	Operating Procedures for the ASR-33 Teletype	3-2
	3.2.1	Reading a Paper Tape (Normal Mode)	3-2
	3.2.2	Duplicating a Paper Tape Off-Line	3-3
	3.3	Manual Bootstrap	3-3
	3.3.1	Operating Procedures for Loading the MANUAL BOOTSTRAP	3-3
	3.3.2	Using the MANUAL BOOTSTRAP to Load a Program Via the ASR-33 Paper Tape Reader	3-5
	3.3.3	Loading the Software LOADER Package Via ASR-33 and MANUAL BOOTSTRAP	3-6

TABLE OF CONTENTS (Continued)

Section			Page
6	CONTINUED		
	6.2.4	Executing a Chain Tape	6-4
	6.2.5	Trace	6-5
	6.2.6	Trace Output	6-5
	6.3	Operator Communications	6-6
7	HARDWARE DIAGNOSTICS		7-1
	7.1	Introduction	7-1
	7.2	810A Mainframe Diagnostic Loading Procedures	7-2
	7.3	Mainframe Exerciser (MFE)	7-6
	7.4	Instruction Simulation and Comparison (IS & C)	7-8
	7.5	Compare Memory to A, A Sign Test (CMASAS)	7-16
	7.6	MEMDEX	7-18
	7.7	Load/Store/Register Change Test (LSRCT)	7-21
	7.8	Arithmetic Test (ADDO)	7-23
	7.9	Multiply Test (MTPY)	7-25
	7.10	Divide	7-27
	7.11	Memory Worst Case Test (MEMTES)	7-29
	7.12	810A Paper Tape Reader/Punch Test	7-31
	7.13	ASR 33/35 Teletype Test	7-34
APPENDIX A	Computer Word Formats		
APPENDIX B	Peripheral Unit Character Codes		
APPENDIX C	Peripheral Unit Command and Test Code Formats		
APPENDIX D	Paper Tape Formats		
APPENDIX E	Assembler Output Formats		
APPENDIX F	Instruction Repertoire		

TABLE OF CONTENTS (Continued)

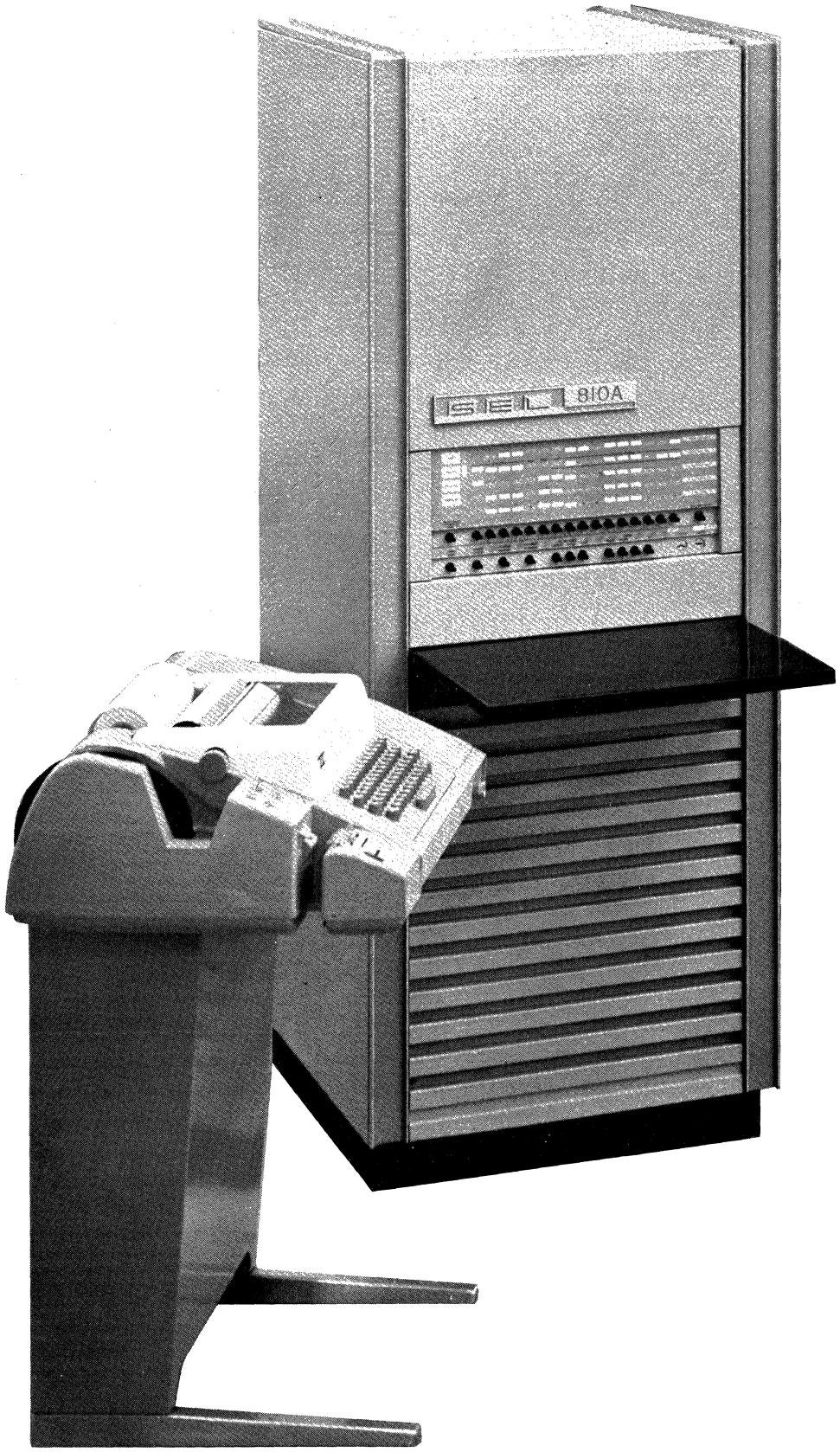
Section			Page
4	SOFTWARE LOADER PACKAGE		4-1
	4.1	Introduction	4-1
	4.2	Operating Procedures	4-1
	4.3	Operator Communications	4-3
	4.3.1	When Loading is Complete	4-3
	4.3.2	EOJ - Typeout	4-3
	4.4	Recovery Procedure	4-3
	4.4.1	"CK" Checksum Error	4-3
	4.4.2	"MO" Memory Overflow	4-4
	4.4.3	Absolute Loader	4-4
	4.5	Update/Debug Utility Procedures	4-4
	4.5.1	Update Instructions and Procedures	4-4
	4.5.2	Debug Instructions and Procedures	4-7
5	SOFTWARE MENEMBLER ASSEMBLER PACKAGE		5-1
	5.1	Introduction	5-1
	5.1.1	Computer Configuration	5-1
	5.1.2	Assembler Modes	5-1
	5.2	Operating Procedures for MNEMBLER Assembler	5-1
	5.2.1	Paper Tape Preparation of Source Program	5-1
	5.2.2	Deletion of Errors on the Source Tape	5-2
	5.2.3	Loading the MNEMBLER Assembler	5-2
	5.2.4	Assembling a Paper Tape Source Program (Typical)	5-3
	5.2.5	Selection of Assembler Options by SENSE Switch Settings	5-4
	5.2.6	Symbolic Listing Format	5-5
	5.3	Operator Communications	5-7
	5.4	Recovery Procedures	5-8
	5.4.1	Table Size	5-8
	5.5	Update/Debug Utility Procedures	5-8
	5.5.1	Update Instructions and Procedures	5-8
	5.5.2	Sense Switch Settings	5-9
	5.5.3	Debug Instructions and Procedures	5-10
6	SOFTWARE FORTRAN IV COMPILER PACKAGE		6-1
	6.1	Introduction	6-1
	6.2	Operating Procedures	6-1
	6.2.1	Executing Compiler Generated Programs	6-3
	6.2.2	Chaining for FORTRAN IV Programs	6-3
	6.2.3	Preparing a Chain Tape	6-4

LIST OF ILLUSTRATIONS

Figure		Page
1-1	SEL 810A Block Diagram	1-1
1-2	SEL 810A Word Format	1-2
2-1	SEL 810A Console Control Panel and Functions	2-3
2-2	Clear the T-REGISTER	2-5
2-3	Load T REGISTER with Example 077277	2-6
2-4	Clear the A ACCUMULATOR	2-7
2-5	Load the A ACCUMULATOR with Example 042000	2-8
2-6	Clear the INSTRUCTION REGISTER	2-9
2-7	Load the INSTRUCTION REGISTER	2-10
2-8	Load the PROGRAM COUNTER with Address 00005	2-11
2-9	Load MEMORY at Address 00005 with Instruction 130101 (CEU)	2-12
2-10	DISPLAY MEMORY at Address 00005	2-13
6-1	A ACCUMULATOR	6-2
6-2	TRACE Listing Output Format	6-6
7-1	Binary Bootstrap for Self-Loading Tapes for ASR-33	7-4
7-2	Binary Bootstrap for Self-Loading Tapes for High Speed Reader	7-5

LIST OF TABLES

Table		Page
3-1	SEL 810A Bootstrap for Use With ASR-33 Reader	3-4
4-1	Keyword Summary	4-13



N3725

SEL 810A General Purpose Computer

SECTION 1

BASIC 810A SYSTEM

1.1 INTRODUCTION

The purpose of this document is to explain the operation of the SEL 810A. Detailed hardware and software procedures required for efficient use of the system are provided.

This document also provides the user with information on the programmer's Console Control Panel and other basic input/output devices.

1.2 BASIC SYSTEM

The 810A Computer (see Figure 1-1) consists of four major units: memory, control, arithmetic and input/output. The memory stores instruction words that define the operation of the computer and data words on which the computer operates. The control unit selects instruction words, decodes them and issues commands to operate the computer. The arithmetic unit performs computations with data words supplied by the input/output unit and the memory unit under the direction of the control unit. The input/output unit transmits data words, commands and status reports between the computer and peripheral equipment. The computer uses 16-bit binary words which are transferred in parallel between the computer units. Arithmetic operations are performed using binary arithmetic with negative words stored in the two's complement form.

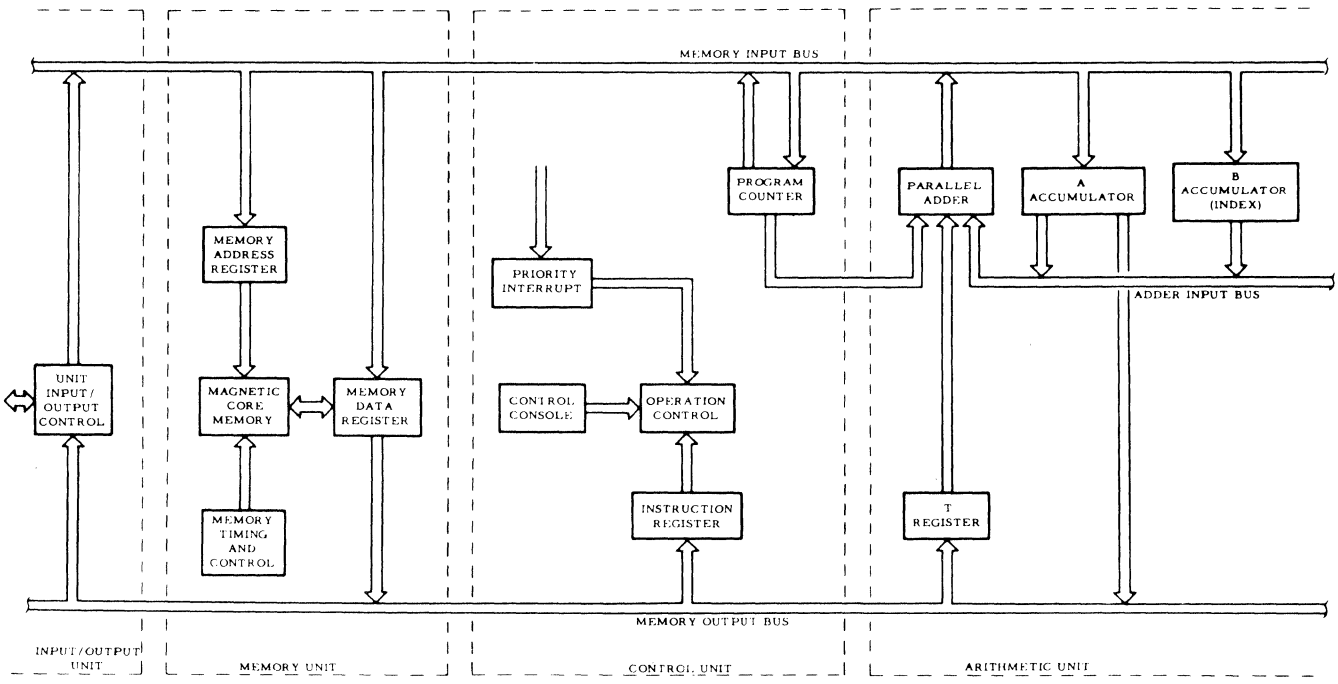


Figure 1-1. SEL 810A Block Diagram

The basic SEL 810A is equipped with an I/O control unit that can connect 64 I/O units to the processor. For maximum reliability 16-bit data words are transferred through the I/O control to and from memory or to and from the A Accumulator. Figure 1-2 shows the SEL 810A Word Format. Up to eight block transfer control units can be added as options to allow direct communication between I/O devices and computer memory.

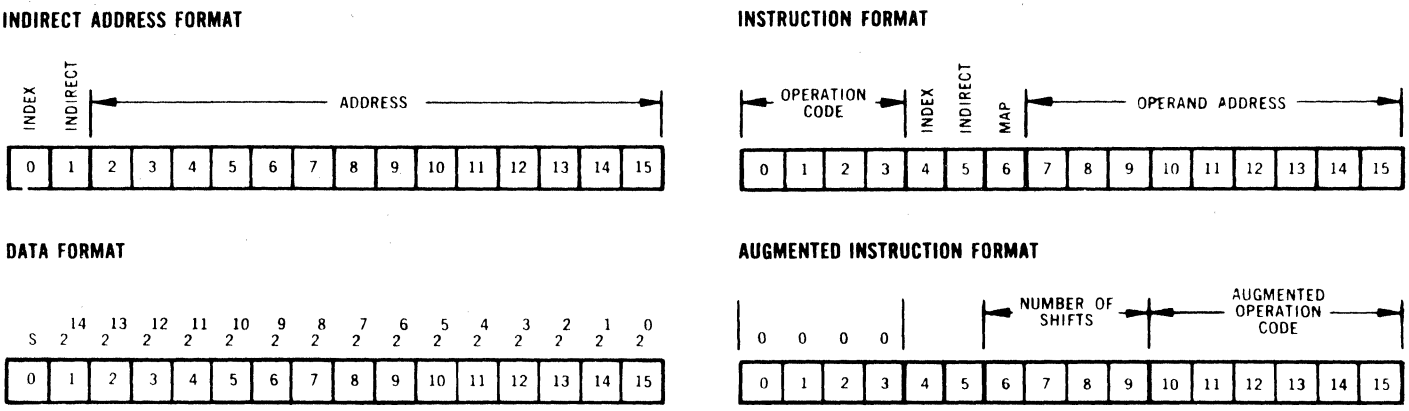


Figure 1-2. SEL 810A Word Format

1.2.1 Specifications

All silicon monolithic integrated logic circuits
16-bit word length
4096 word memory
1.75 microsecond full cycle time
Full parallel operation
Computation time including access and indexing

Add, Subtract	3.5 microseconds
Multiply	7.0 microseconds
Divide	10.5 microseconds

Double-length accumulator
Hardware index register (B Accumulator)
I/O structure capable of handling 64 units or controllers (Drivers and terminators for 16 units supplied with the basic computer)
Two separate levels of priority interrupts

Four sense switches.
Switch-addressable program halt.
Power fail safe.

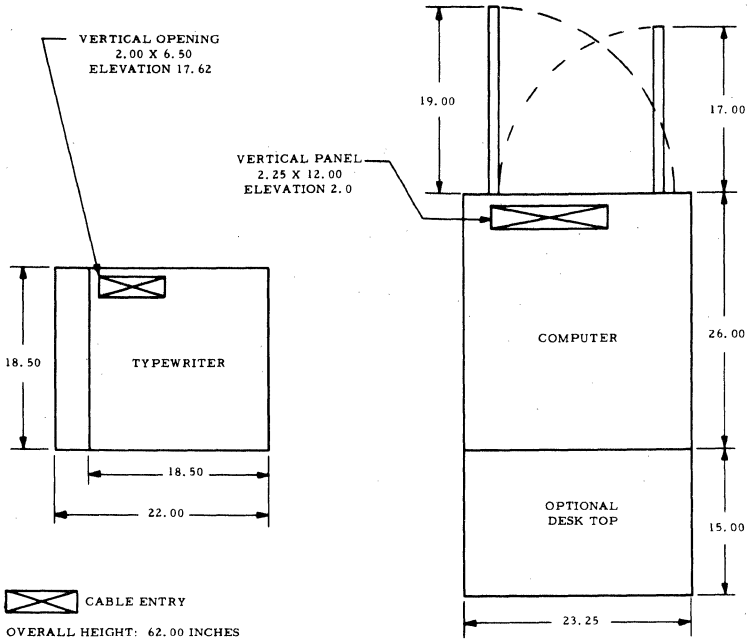
1.2.2 Options

Basic I/O Unit: ASR-33 or ASR-35 typewriter with paper tape reader and punch.
Memory expandable to 32K (4K and 8K modules available).
Memory parity bit with parity generator/checker.
Program protect bit for guarding individual locations.
Up to 96 individual levels of priority interrupts.
Variable base register - increases direct addressing capability.
Instruction trap - can prevent execution of privileged instructions.
Stall alarm.
Power fail safe, data store and restore.
Up to eight fully buffered block transfer channels.

1.2.3 Standard Software

Full ASA FORTRAN compiler - (8K min).
FORTRAN Library.
MNEMBLER assembler - one or two pass, relocatable object format.
Compiler/assembler loader.
Utility routines - debugging aids, I/O handlers, tape editor.
Maintenance routines - complete set for computer and peripheral units.

1.3 INSTALLATION REQUIREMENTS



POWER

Power - 1,200 watts (basic computer)
Voltage - 115 VAC ±10%, 60 cps ±1%, 1 single phase.

HEAT DISSIPATION & COOLING

BTU/ - 4,100
Integral blower included.

ENVIRONMENT

Temperature (Storage)	0° to 150° F
Temperature (Operating)	50° to 95° F
Relative Humidity	30% to 90%

ACCESS

Logic - Rear
Circuit Breaker - Front
Connectors
(a) AC - Rear
(b) Data - Rear

CABLES

Power - 3 wire, Hubbel Type 7313 supplied-mates with 7327 receptacle.

Data
(a) Type - SEL 80-060A
SEL 80-061A
SEL 80-062A
(b) Number - Depends on installation.

CIRCUIT BREAKER

Rating - 15 amp

WEIGHT

300 pounds

SECTION 2

CONSOLE OPERATIONS

2.1 INTRODUCTION

The SEL 810A is supplied with two modes of operator communication: the Console Keyboard Printer (ASR-33 Teletype) and the Console Control Panel. The Console Control Panel offers the operator a means of inspecting, changing, and controlling the functions of hardware and software.

2.2 CONSOLE INPUT/OUTPUT

2.2.1 ASR-33 Teletype Unit (Standard)

The ASR-33 Teletype Unit is the standard I/O device provided to communicate with the SEL 810A in the on-line mode. The ASR -33 is a versatile device providing a capability to read paper tape at 20 characters per second or punch paper tape at 10 characters/second. The ASR-33 also prints out computer or transfer data at 10 characters/second between the SEL 810A and the keyboard. In the local mode the unit is used for off-line paper tape preparation, reproduction, and listing. In the on-line mode it is used for computer input from the console keyboard or paper tape reader and for computer output to the console printer and paper tape punch. The input devices and output devices operate independently in the on-line mode.

2.2.2 Keyboard Printer

The ASR -33 keyboard, similar to that of a standard typewriter, includes four rows of keys and generates an eight-level code. Letters and numerals are transmitted without a shift, similar to lower-case transmission on a typewriter. A few special characters (?, =, *, etc.) are typed by using the Shift Key, similar to upper-case positions on certain typewriter keys. The keyboard activates the printer in the local mode only. If printing is desired in the on-line mode, each input character is output to the printer under program control. The keyboard is enabled by an appropriate CEU instruction prior to use in the on-line mode.

2.3 PRINTER OPERATION

The printer will only accept 67 different ASCII characters:

(1) Letters of the alphabet

(2) Numerals

(3) 28 Special characters

(4) Carriage return, line feed, bell

2.3.1 Local Operation

Local operation of the ASR-33 includes the following data transmission:

(1) Keyboard to printer

(2) Reader to printer

2.3.2 On-Line Operation

The printer is used in the on-line mode by executing the appropriate output instructions under program control.

2.3.3 Keyboard Interlock

The ASR-33 keyboard is interlocked when any key is depressed except the SHIFT, CTRL or REPT keys, preventing more than one key from being depressed at a time. The keyboard does not lock in the upper-case position. Therefore, the operator must hold the SHIFT key depressed to produce upper-case characters.

2.4 PAPER TAPE READER

The reader is started under program control by executing a CEU instruction with the appropriate function data.

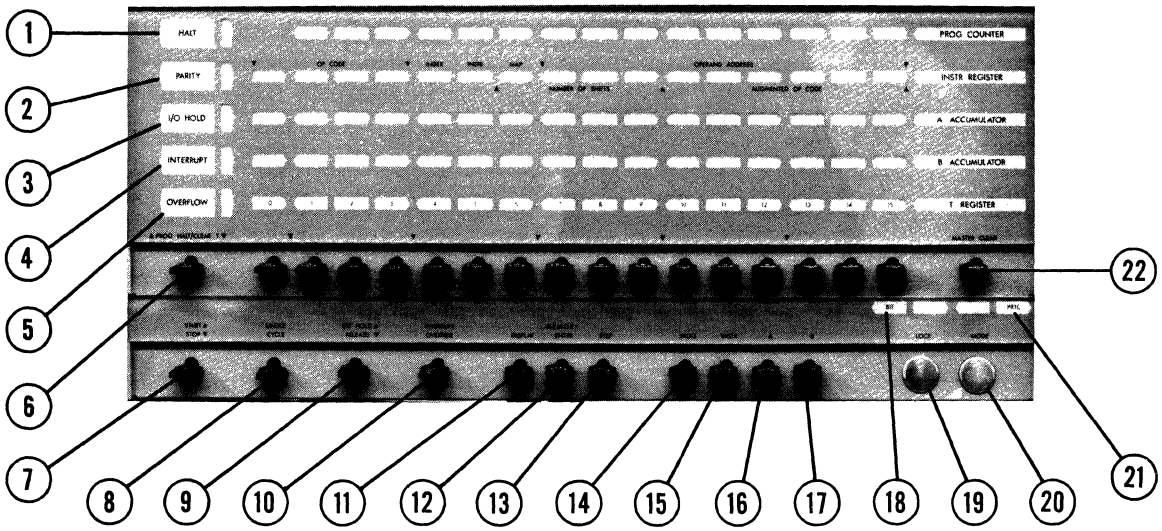
Local starting is controlled by the START/STOP switch. When the reader is started, the first character to be read is the one initially positioned over the read pins.

2.5 PAPER TAPE PUNCH

The punch is controlled by manual operation of the punch ON-OFF switch located on the ASR-33. When the punch is on, any output to the ASR-33 printer causes tape to be punched. Any character is punched whether it is printable or not.

2.6 CONSOLE OPERATIONS

Manual control of the computer is performed by manipulation of toggle switches on the Console Control Panel. Indicators and toggle switches located on the panel face are used to monitor and alter programs. A group of 16 double throw, dual purpose toggle switches allows entry of data. These toggle switches are used to manually enter or modify the contents of computer memory, such as loading the manual bootstrap. The identification and functions of the Console Control Panel are shown in Figure 2-1.



1 Lights to indicate a program halt.

2 Lights to indicate the detection of a memory parity error. The parity error indicator will be reset when the start/stop switch is depressed.

3 Lights to indicate a wait for I/O function.

4 Lights to indicate a priority interrupt.

5 Lights to indicate an arithmetic condition.

6 Raised to connect switches 0-15 as program HALT switches. Depressed to clear the contents of the "T" register.

7 Depressed once to start program. The next time depressed will stop the program.

8 Depressed to execute single instructions in normal sequence.

9 Depressed to release I/O wait and allow computer to resume.

10 Depressed to inhibit operation of priority interrupts (lock).

11 Depressed to display the contents of the current memory location.

12 Depressed to load the contents of T into the current memory location.

13 Used with the display or enter switch (in up position) to display or enter sequential memory locations by depressing the step switch.

14 Depressed to transfer the contents of the T register to the program counter.

15 Depressed to transfer the contents of the T register to the instruction register.

16 Depressed to transfer the contents of the T register to the A accumulator.

17 Depressed to transfer the contents of the T register to the B accumulator.

18 *Indicates the presence of a program protect bit in the latest word accessed from memory.

19 *This key lock switch inhibits the operation of all control switches.

20 *This key lock switch puts the computer in the program protect mode.

21 *Indicates the status of the program protect latch.

22 Depressed to clear all major registers and control latches.

*These indicators and switches are supplied with the program protect option.

N3924

Figure 2-1. SEL 810A Console Control Panel and Functions

Several registers are affected by console operation. These registers are affected only while the computer is in a HALT condition or during the execution of the special instructions Load Control Switches (LCS) and Sense Numbered Switch (SNS).

(1) The Instruction Register holds and interprets the instruction to be executed.

(2) The Program Counter holds the address of the instruction to be executed.

(3) The A Accumulator is the primary arithmetic register.

(4) The B Accumulator holds the multiplier during multiply operations and stores the least significant bits of the product.

(5) The T Register is the intermediate storage between the 16 entry toggles and the other registers. The T Register is used to facilitate efficient use of the console as an operations monitor.

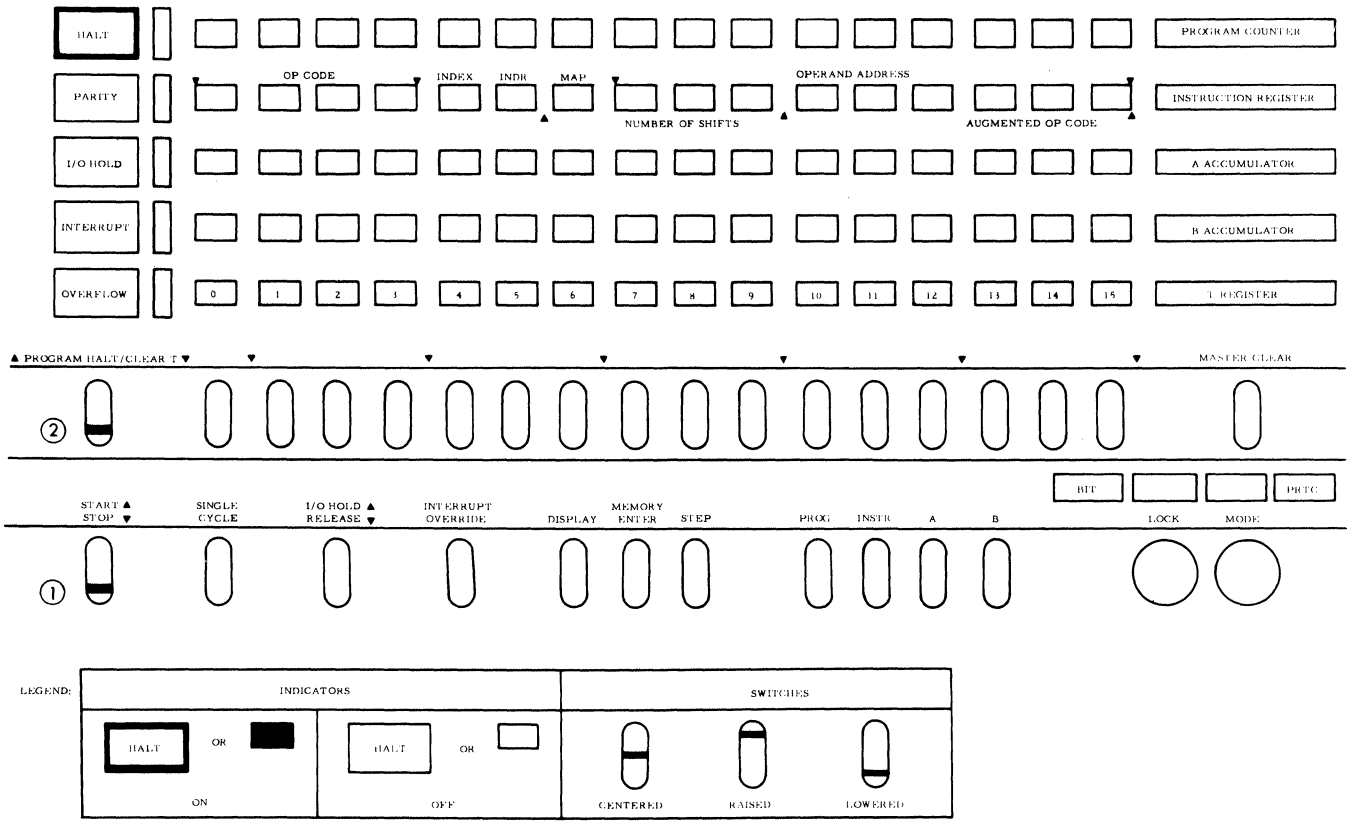
The T Register is the interface between the 16 entry toggles and the computer's working arithmetic and control registers.

NOTE

The HALT and I/O HOLD RELEASE toggle switches are the only functioning controls after the START toggle switch is depressed. Parity errors and I/O Interrupts can only be reset while the computer is in a HALT state.

2.6.1 Example Console Operations

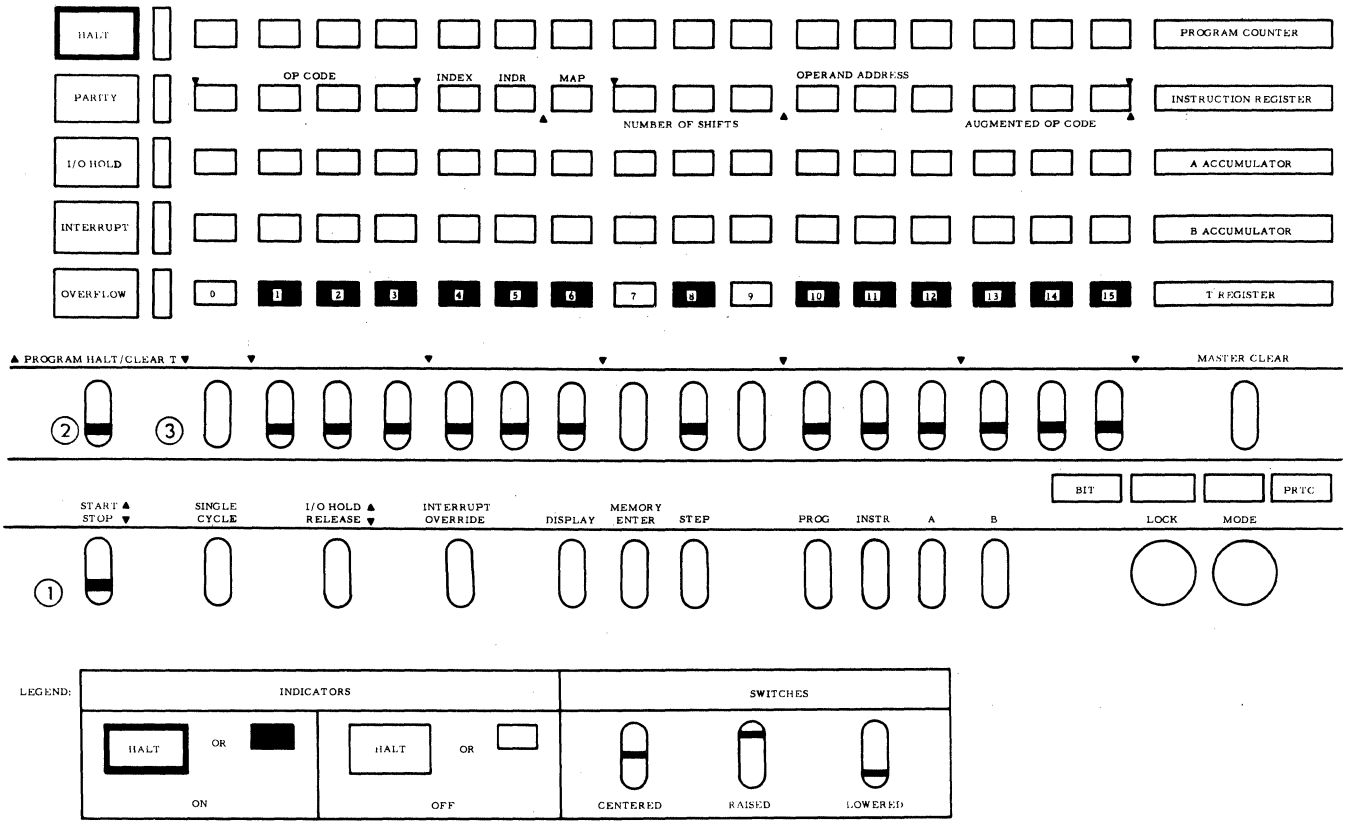
The following examples, Figures 2-2 through 2-10, demonstrates the loading and clearing of the T Register, A Accumulator, Instruction Register, Program Counter, and Memory. Only those switches moved or changed from neutral are shown for clarity of presentation. The indicator lights are presented in the same manner. Data is expressed in octal notations, bits 13, 14, and 15 are the least significant octal character (8^0) in the examples shown. Bits 1, 2, and 3 are the most significant octal character (8^4). Bit 0 is the sign bit.



Step 1. Depress START/STOP toggle switch.

Step 2. Depress PROG HALT/CLEAR T toggle switch. Note that the T REGISTER has been set to zero.

Figure 2-2. Clear the T-REGISTER

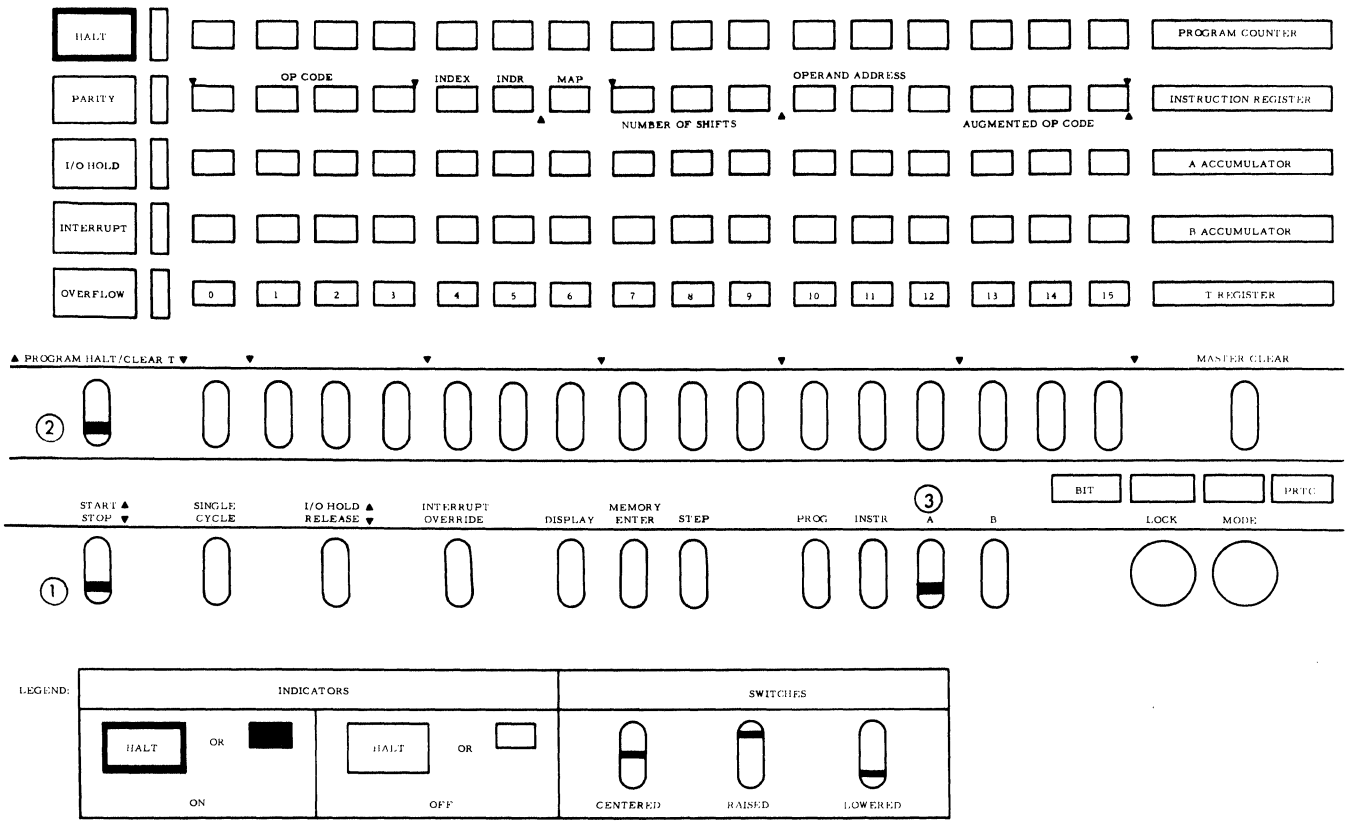


Step 1. Depress START/STOP toggle switch.

Step 2. Depress PROG HALT/CLEAR T toggle switch.

Step 3. Depress switches 0-15 according to the data to be loaded (077277). Note that the indicators for the T-REGISTER read 077277.

Figure 2-3. Load T REGISTER with Example 077277

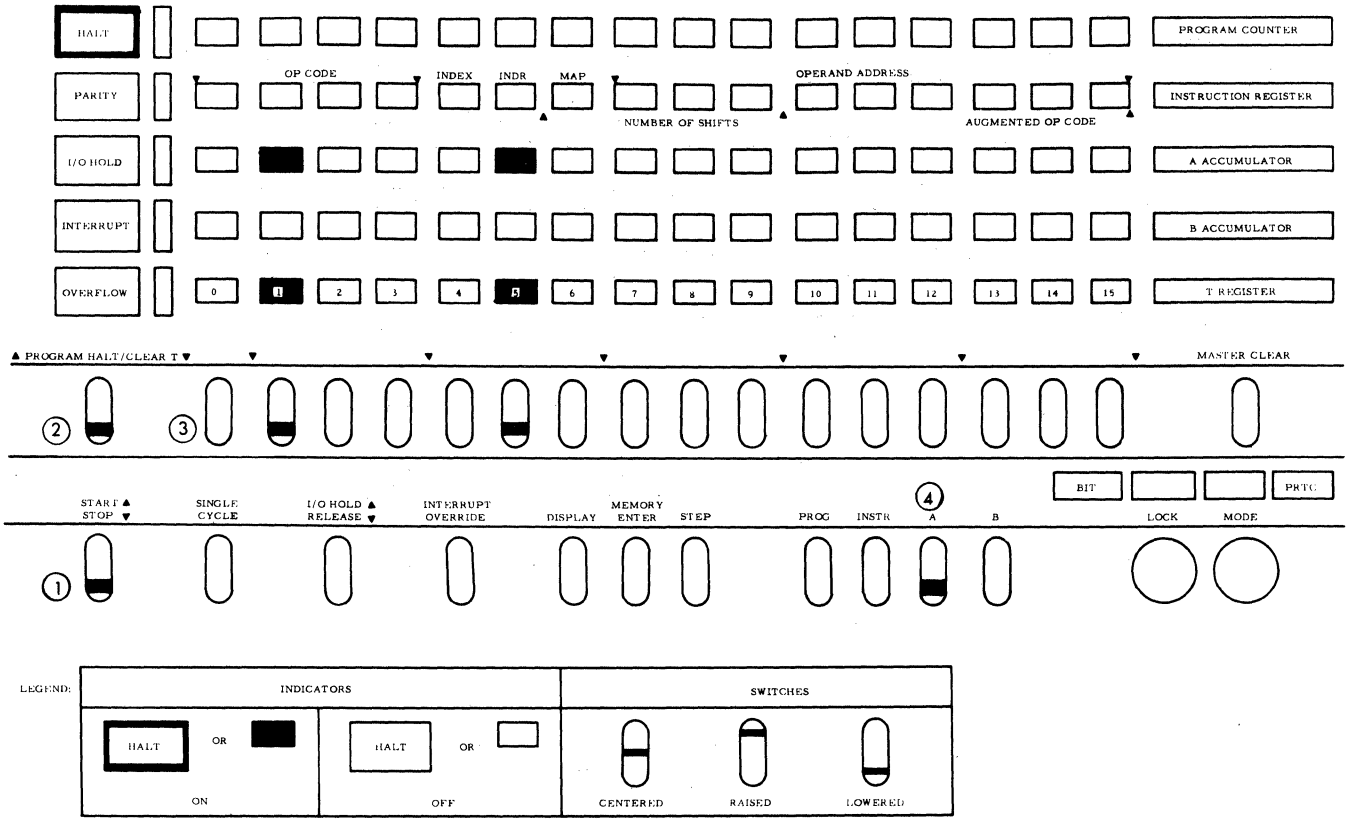


Step 1. Depress START/STOP toggle switch.

Step 2. Depress the PROG HALT/CLEAR T toggle switch.

Step 3. Depress the INSTR toggle switch. Note the zero condition of the A ACCUMULATOR.

Figure 2-4. Clear the A ACCUMULATOR



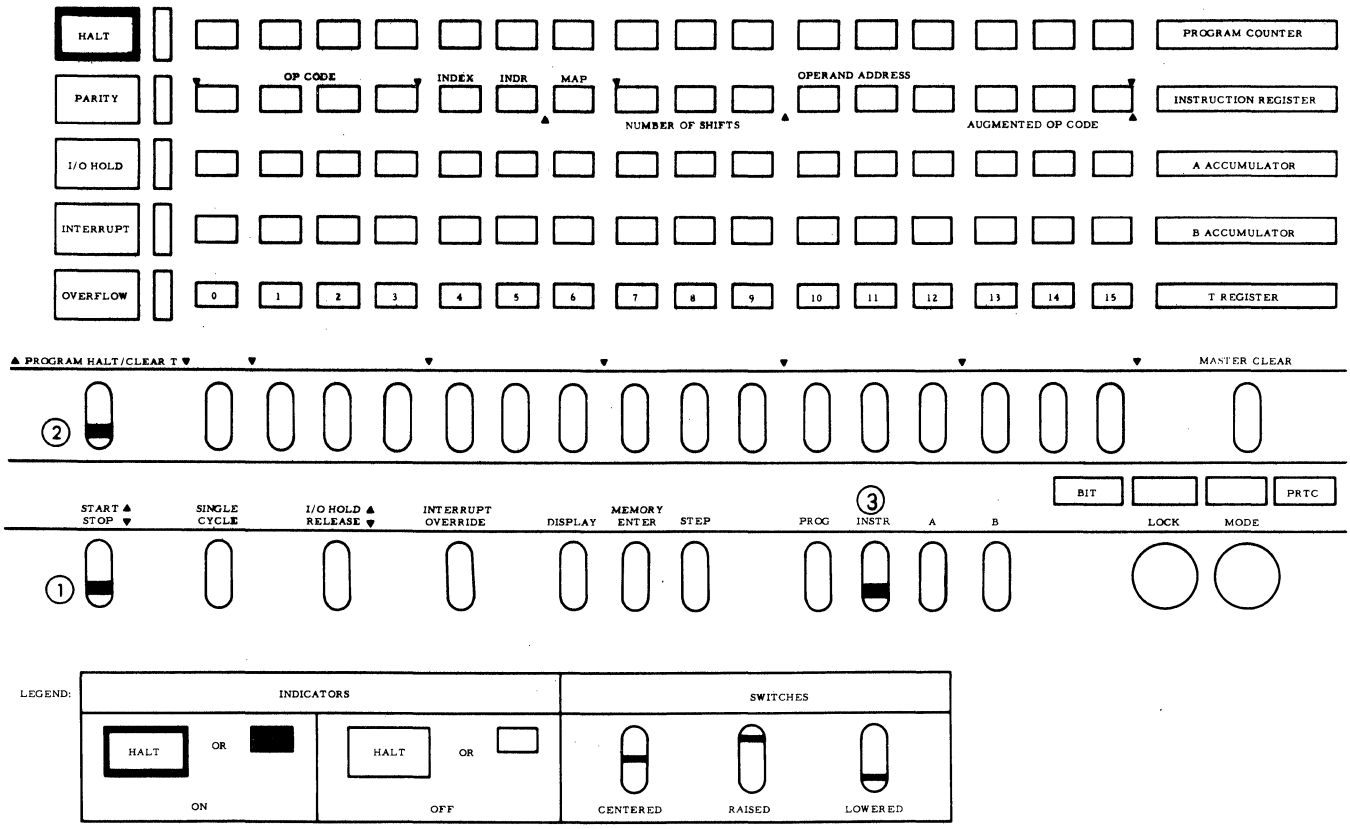
Step 1. Depress START/STOP toggle switch.

Step 2. Depress the PROG HALT/CLEAR T toggle switch.

Step 3. Depress switches 0-15 according to the data to be loaded (Example - 042000).

Step 4. Depress A toggle switch. A ACCUMULATOR indicators should read 042000.

Figure 2-5. Load the A ACCUMULATOR with Example 042000

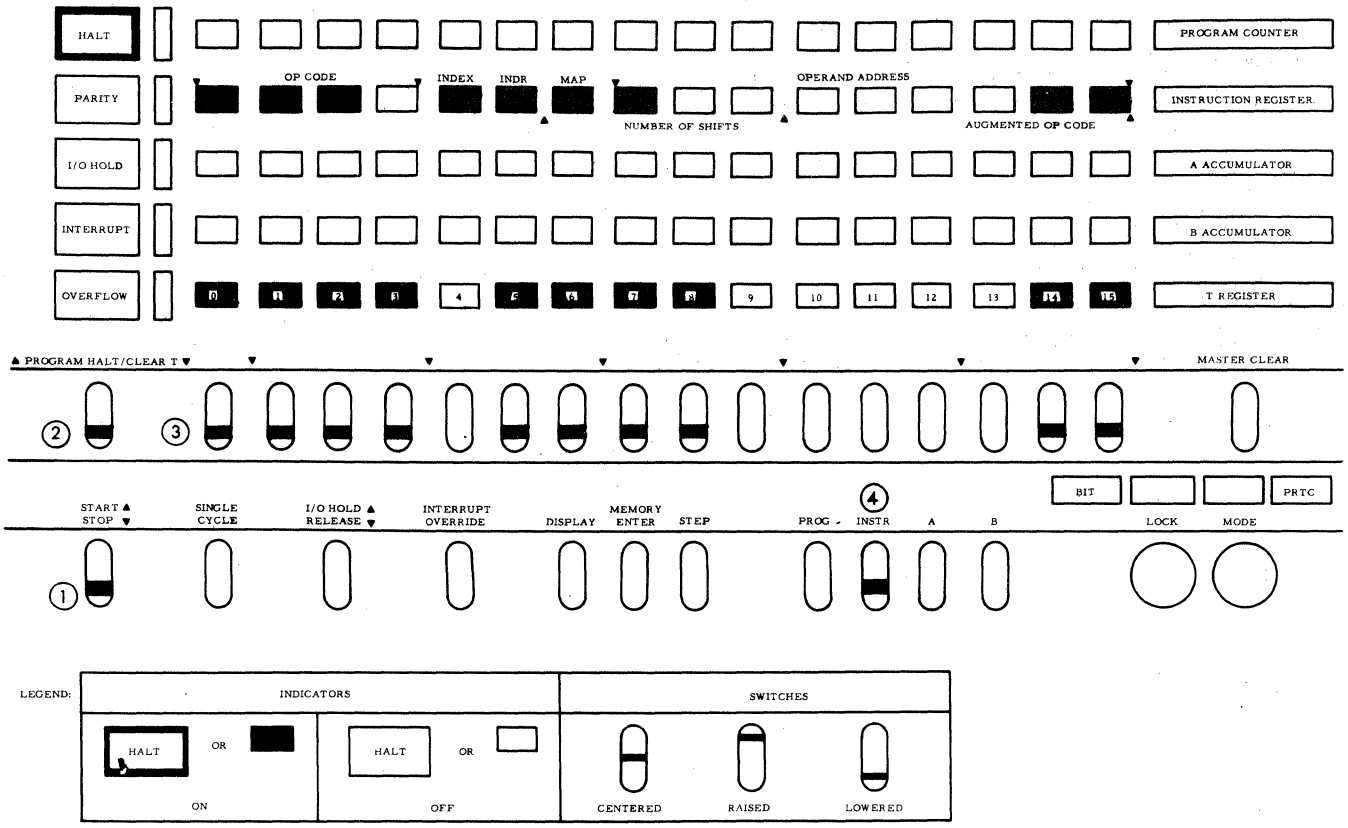


Step 1. Depress START/STOP toggle switch.

Step 2. Depress the PROG HALT/CLEAR T toggle switch.

Step 3. Depress the INSTR toggle switch. Note the indicators in the INSTRUCTION REGISTER read 000000 .

Figure 2-6. Clear the INSTRUCTION REGISTER



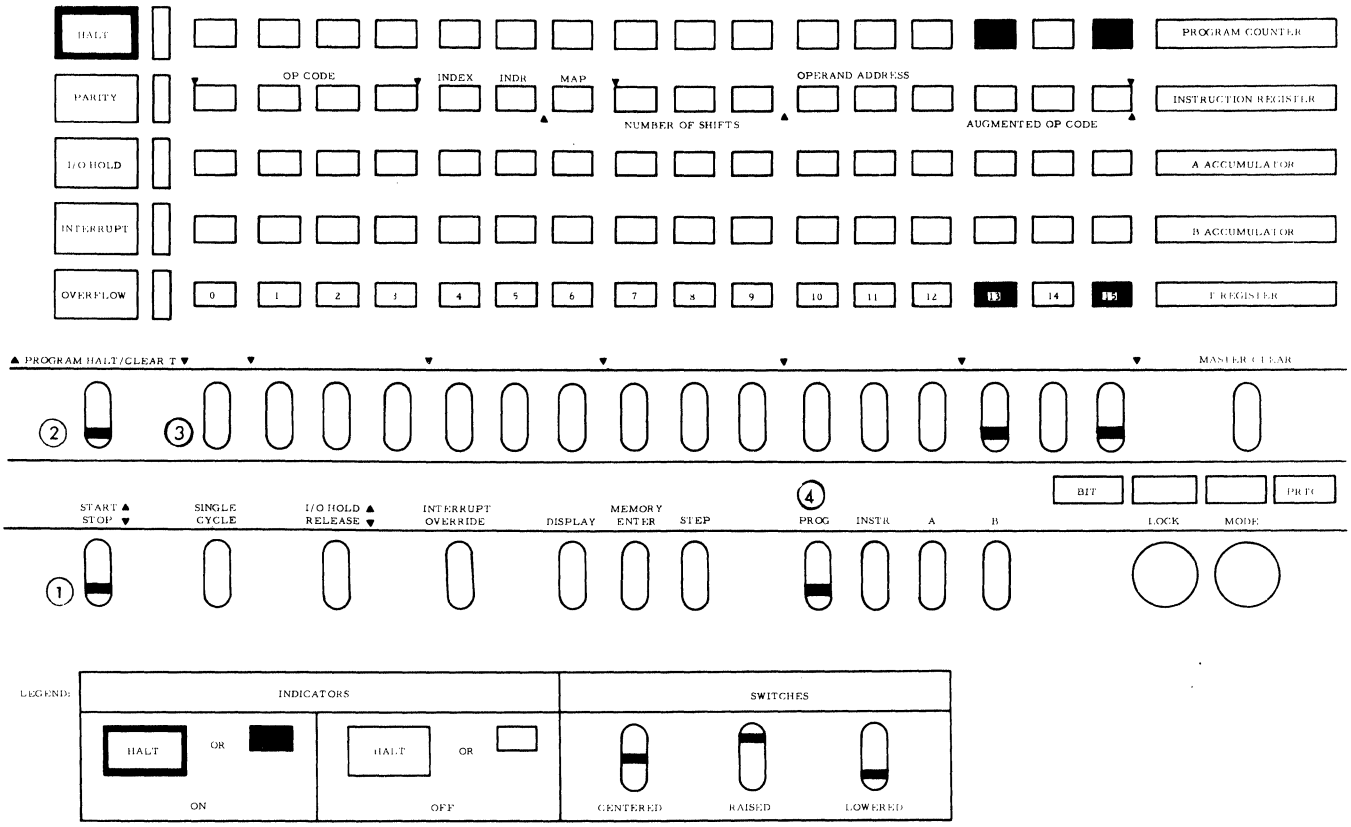
Step 1. Depress START/STOP toggle switch.

Step 2. Depress the PROG HALT/CLEAR T toggle switch.

Step 3. Depress the switches 0-15 to enter the data to be loaded (173603).

Step 4. Depress the INSTR toggle switch. INSTRUCTION REGISTER indicators should display 173603.

Figure 2-7. Load the INSTRUCTION REGISTER



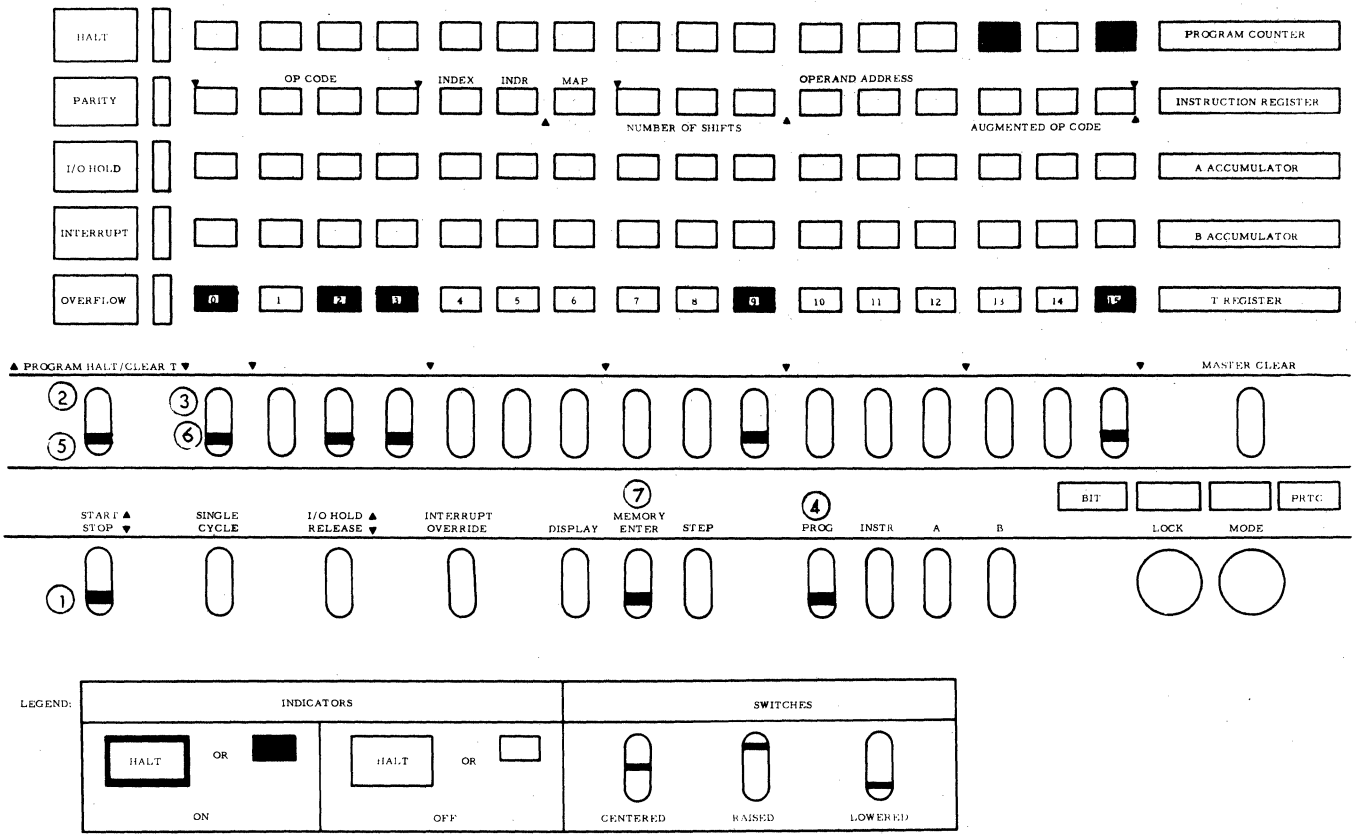
Step 1. Depress START/STOP toggle switch.

Step 2. Depress PROG HALT/CLEAR T toggle switch.

Step 3. Depress the 0-15 switches to enter the address (00005).

Step 4. Depress the PROG toggle switch. PROG COUNTER indicators should display 00005.

Figure 2-8. Load the PROGRAM COUNTER with Address 00005



Step 1. Depress START/STOP toggle switch.

Step 2. Depress PROG HALT/CLEAR T toggle switch.

Step 3. Depress the 0-15 switches to select address 00005.

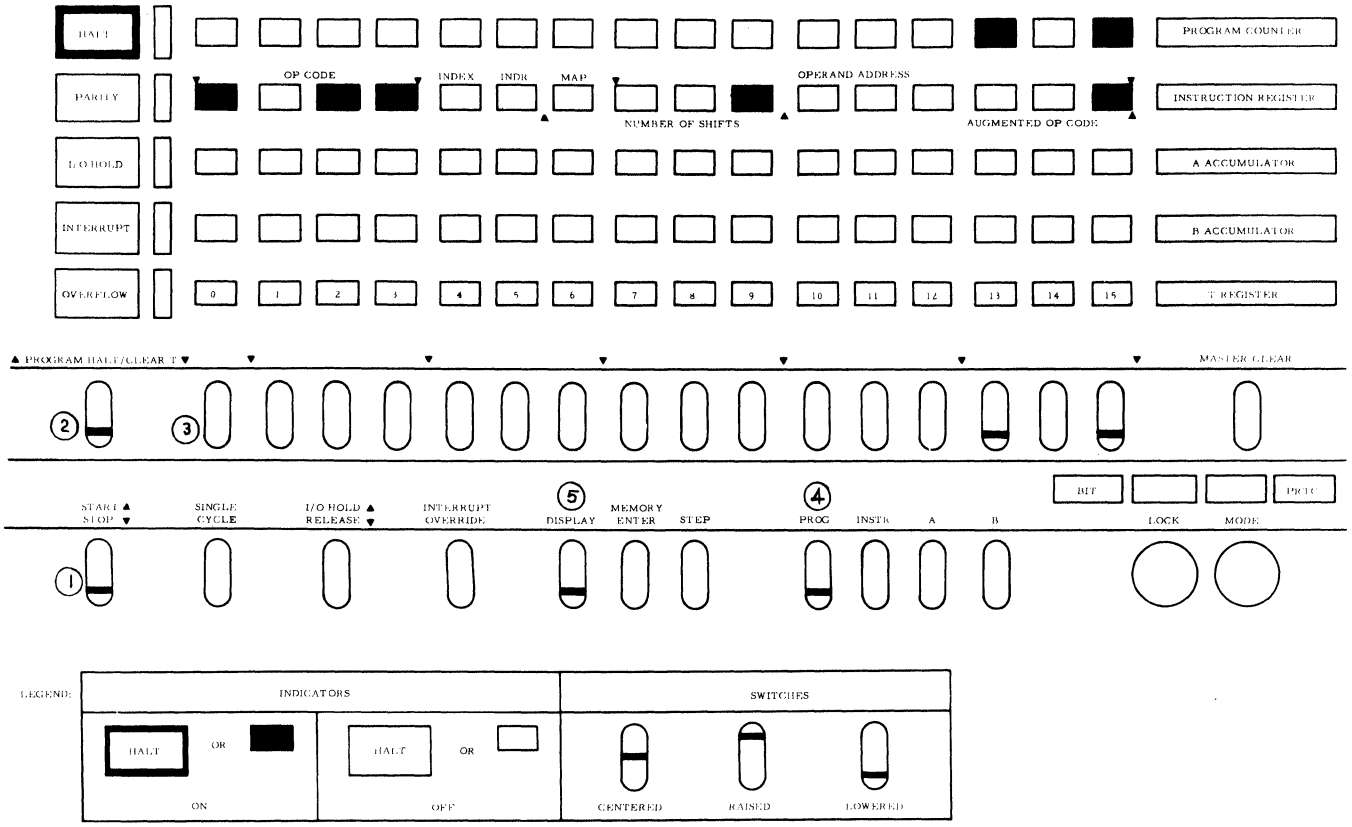
Step 4. Depress PROG toggle switch.

Step 5. Depress PROG HALT/CLEAR T toggle switch.

Step 6. Depress 0-15 switches to select data 130101.

Step 7. Depress MEMORY ENTER toggle switch.

Figure 2-9. Load MEMORY at Address 00005 with Instruction 130101 (CEU)



Step 1. Depress START/STOP toggle switch.

Step 2. Depress PROG HALT/CLEAR T toggle switch.

Step 3. Depress the 0-15 switches to select address 00005.

Step 4. Depress PROG toggle switch to load program counter.

Step 5. Depress MEMORY DISPLAY switch to display contents of address 00005 on INSTRUCTION REGISTER indicators. The instruction register should contain 130101 from procedure in Figure 2-9.

Figure 2-10. DISPLAY MEMORY at Address 00005

SECTION 3

BASIC OPERATING PROCEDURES

3.1 CONSOLE CONTROL PANEL OPERATIONS

The basic operating procedures outlined in this section are for the following:

(1) Normal Computer turn-on.

(2) Normal Computer turn-off.

(3) Mode Selection (Normal or Single Cycle).

3.1.1 Normal Computer Turn-On

Normal Computer turn-on operating procedures assume that programs are already stored in memory and that Computer Power switch is in the OFF position.

Step 1 The computer Power Switch is behind a small door at the top of the front loovered panel. Raise the Computer Power Switch to energize the system.

Step 2. Note the ON condition of the Console Control Panel indicators.

Step 3. Depress the MASTER CLEAR Toggle Switch.

Step 4. Enter the Program Starting Address into the program counter. Refer to Figure 2-8.

Step 5. Depress the START toggle switch to begin program execution.

3.1.2 Normal Computer Turn-Off

Step 1 Depress START/STOP toggle switch.

Step 2 Depress the Computer Power Switch located behind the front loovered panel.

3.1.3 Mode Selection

There are two basic operation modes, Normal and Single Cycle. Normal mode is used for execution of a program once it is stored in memory. Single Cycle is used for loading the manual bootstrap, inspecting or changing memory, and for setting SENSE toggle switches or program halt conditions.

3.2 OPERATING PROCEDURES FOR THE ASR-33 TELETYPE

3.2.1 Reading a Paper Tape (Normal Mode)

Step 1 Locate the paper tape reader and punch unit on the left hand side of the Teletype keyboard.

Step 2 Release the paper tape hold down tab on the reader assembly and insert a paper tape with its leading edge toward the front of the reader unit. The sprocket wheel should correspond to the sprocket punches on the paper tape being read.

Step 3 Place the START, STOP, FREE key in the START position.

Step 4 Locate the Teletype On Line/Off/Off Line switch on the right hand side of the keyboard. If computer operation is desired, use Step 5. If off-line operation is desired (printing or tape copy), use Step 9.

Step 5 Rotate Teletype On Line/Off/Off Line switch to the ON LINE position.

Step 6 Master CLEAR computer and set PROGRAM COUNTER to starting address. Refer to Figure 2-8.

Step 7 Depress the START toggle on the computer Console Control Panel. If the MANUAL BOOTSTRAP was loaded, the paper tape reader will be energized.

Step 8 OFF-LINE operations are handled by rotating the Teletype On Line/Off/Off Line switch to the OFF LINE position.

Step 9 The data on the tape is printed on the keyboard printer.

Step 10 — After the tape is read, turn the START, STOP, FREE switch to the STOP position.

3.2.2 Duplicating a Paper Tape Off-Line

Step 1 — Advance a few inches of paper tape as a leader by depressing the HERE IS key on the Console Keyboard.

Step 2 — Insert the pre-punched paper tape correctly in the paper tape reader unit (see procedure: Reading a Paper Tape).

Step 3 — Activate ON switch located above punch unit.

Step 4 — Follow the procedures for reading a paper tape.

3.3 MANUAL BOOTSTRAP

The function of the MANUAL BOOTSTRAP (see Table 3-1) is to enable the operator to load the ABSOLUTE LOADER PROGRAM. The MANUAL BOOTSTRAP is ONLY entered into memory by operator manipulation of the Console Control Panel toggle switches and indicators (Figures 2-1, 2-8, 2-9, and 2-10). The MANUAL BOOTSTRAP is entered into memory by using the:

(1) Entry Toggles

(2) T Register

(3) Program Counter

(4) Load Memory Toggle Switch

(5) Single Cycle Toggle Switch

The MANUAL BOOTSTRAP does not generate a checksum and is used only to load a program capable of performing such self-checking features. The operator may load the ABSOLUTE LOADER PROGRAM which calculates a checksum for other programs being loaded.

3.3.1 Operating Procedures for Loading the MANUAL BOOTSTRAP

Step 1 — Turn the computer power ON (refer to normal operating procedures 3.1.1).

Step 2 — Depress the MASTER CLEAR toggle switch.

Step 3 — Enter starting address into program counter and first instruction of MANUAL BOOTSTRAP into T-REGISTER.

Table 3-1

SEL 810A Bootstrap for Use with ASR-33 Reader

LOC.	OPER	ADDRESS, INDEX	OCTAL LOC	CODING
STRT	CEU	1, W	0	130101
	DATA	'4000	1	004000
	AIP	1, W	2	170301
	SAZ		3	000022
	BRU	*+2	4	111006
	BRU	*-3	5	111002
READ	AIP	1, W	6	170301
	LSL	8	7	001016
	AIP	1, W, R	10	174301
	STA*	DAC 1	11	033016
	SAZ		12	000022
	IBS		13	000026
	BRU	DAC 2	14	113017
	BRU	READ	15	111006
DAC 1	DAC	CHAN-2, 1	16	Note 1
DAC 2	DAC	CHAN	17	Note 2

NOTE 1 8K - 117671 16K - 137671

NOTE 2 8K - 17673 16K - 37673

Step 4 Raise MEMORY ENTER switch.

Step 5 Depress STEP switch.

Step 6 Enter second instruction into T register.

Step 7 Depress STEP switch.

Step 8 Repeat Steps 6 and 7 for remaining MANUAL BOOTSTRAP instructions.

Step 9 Center MEMORY ENTER switch.

Step 10 Depress MASTER CLEAR switch.

Assembler and Compiler generated paper tapes are in relocatable i.e., relative address, format. Relocatable tapes are loaded by using one of the following methods:

(1) Software LOADER Program

(2) Software Utility Programs

Conversion to non-relocatable format is performed by using the Absolute LOAD/DUMP Programs or PAPER TAPE DUMP Package in the DEBUG Program.

It is recommended that each system having a binary non-relocatable paper tape of the relocatable LOADER Program. This allows, at start-up time or whenever needed, the reading of the LOADER PROGRAM into memory by using the MANUAL BOOTSTRAP. This procedure enables any relocatable paper tape to be read into core memory, including the ASSEMBLER and COMPILER Programs.

3.3.2 Using the MANUAL BOOTSTRAP to Load a Program Via the ASR-33 Paper Tape Reader

Step 1 With the MANUAL BOOTSTRAP loaded, depress the MASTER CLEAR toggle switch.

Step 2 No SENSE switches are required for use of the MANUAL BOOTSTRAP.

Step 3 Load the starting address of the MANUAL BOOTSTRAP in the PROGRAM COUNTER.

Step 4 Clear the T REGISTER.

Step 5 — Place the binary formatted program tape on the PAPER TAPE READER.

Step 6 — Ready the Console Keyboard Printer by turning the On Line/Off/ Off Line switch to the ON LINE position.

Step 7 — Ready the paper tape reader by setting the START, STOP, FREE switch to the START position.

Step 8 — Depress the START toggle switch on the Computer Console Control Panel.

Step 9 — The reader arrives at a normal HALT when a STOP CODE is read.

Abnormal Conditions:

Step 1 — Check for correct loading of the MANUAL BOOTSTRAP if the above procedure fails. Use the DISPLAY MEMORY example (Figure 2-10) for checking the contents of memory.

Step 2 — Reload the MANUAL BOOTSTRAP if necessary.

Program Starting Procedure:

Step 1. — Depress the MASTER CLEAR toggle switch.

Step 2 — Load the program starting address into the PROGRAM COUNTER.

Step 3 — Depress the Computer START toggle switch.

3.3.3 Loading the Software LOADER Package Via ASR-33 and MANUAL BOOTSTRAP

Step 1 — With the MANUAL BOOTSTRAP loaded, depress the MASTER CLEAR toggle switch.

Step 2 — No SENSE switches are required for use of the MANUAL BOOT-STRAP.

Step 3 — Load the starting address of the MANUAL BOOTSTRAP in the PROGRAM COUNTER.

Step 4 — Clear the T REGISTER.

Step 5 — Place the binary non-relocatable version of the Software LOADER Package in the Paper Tape Reader adjacent to the Console Key-board Printer.

Step 6 — Ready the Console Keyboard Printer by rotating the On Line/Off/Off Line switch to the ON LINE position.

Step 7 — Ready the paper tape reader by setting the START, STOP, FREE switch to the START position.

Step 8 — Depress the START toggle switch on the Computer Control Panel.

NOTE

Operation will arrive at a normal HALT when a STOP CODE is recognized.

Abnormal Conditions:

Step 1 — Check for correct loading of the MANUAL BOOTSTRAP if the above procedure fails. Use the DISPLAY MEMORY example (Figure 2-10) for checking the contents of memory.

Step 2 — Reload the MANUAL BOOTSTRAP if necessary.

Program Starting Procedure:

Step 1 — Depress the MASTER CLEAR toggle switch.

Step 2 — Load the program starting address into the PROGRAM COUNTER.

Step 3 — Depress the Computer START toggle switch.

SECTION 4

SOFTWARE LOADER PACKAGE

4.1 INTRODUCTION

The SEL 810A Object Program Loader (Catalog No. 300001A) is designed to be compatible with the 810A MNEMBLER Assembler and the FORTRAN IV Compiler.

The Loader provides for relocatable and absolute instructions. The capability of using pre-compiled subroutine libraries is included in a manner which guarantees that a given routine is only loaded once, regardless of the order of the library.

Loading and chaining, if requested, is done using modular input/output driver subroutines, allowing maximum flexibility in choice of loading hardware devices.

The Loader loads and completes the linkage between the main program and the subroutines.

4.2 OPERATING PROCEDURES

This procedure assumes that the Manual Bootstrap and Loader programs have been read into memory according to Section 3.3.

NOTE

The LOADER uses the ENTRY TOGGLES as SENSE switches. SENSE switches are the ENTRY TOGGLES in a raised state. Sense/Halt switch must be in position SENSE for sense switch to be effective.

The main program must be loaded first. Once the main program is loaded, the subroutine library may be loaded for automatic integration by the software LOADER. These steps are as follows:

Step 1 — Depress START/STOP toggle on the Console Control Panel to stop the computer.

Step 2 — Depress computer MASTER CLEAR toggle.

Step 3 — Place the program to be loaded in the Paper Tape Reader.

Step 4 — Rotate the Teletype On Line/Off/Off Line switch to the ON LINE position.

Step 5 — Ready the paper tape reader by setting the START, STOP, FREE switch to the START position.

Step 6 — Set the PROGRAM COUNTER to the starting address for LOADER (refer to Figure 2-8, Loading PROGRAM COUNTER).

Step 7 — Set the A Accumulator to the program starting address.

NOTE

a) A Accumulator is set to zero if the program is absolute.

b) A Accumulator is set to the relocation base if the program is relative.

Step 8 — Set the B Accumulator to the MAP starting address. MAP must be greater than 10 if library routines are to be called.

NOTE

B Accumulator is set to 777 if the MNEMBLER assembler is being loaded into MAP zero.

Step 9 — Set SENSE switches by raising the ENTRY TOGGLES to the desired SENSE switch setting according to the following table.

ENTRY TOGGLE	STATE	FUNCTIONS TO BE PERFORMED
0	Raised	INPUT ACCEPTED FROM OPTIONAL HIGH SPEED PAPER TAPE READER.
0	Neutral	INPUT ACCEPTED FROM ASR-33 PAPER TAPE READER.
1	Raised	PRINTED OUTPUT WILL LIST ALL SUBROUTINES ON THE KEYBOARD PRINTER.
1	Neutral	WILL NOT LIST SUBROUTINES.
2	Raised	WILL LIST ALL SUBROUTINES NOT LOADED.
2	Neutral	NO PRINTED OUTPUT.
3	Raised	INPUT FROM MAGNETIC TAPE UNIT.

Step 10 — Depress START toggle switch.

4.3 OPERATOR COMMUNICATIONS

4.3.1 When Loading is Complete

(1) If SENSE switch No. 2 is RAISED, the typeout includes a list of subroutines that are missing.

(2) If SENSE switch No. 1 is RAISED, the typeout includes a list of all the subroutines loaded.

(3) If no subroutines are required, the typeout contains "LCEJ" and the program arrives at a normal HALT.

(4) The LOADER will HALT after each library paper tape load. Depressing START when more subroutines are needed causes the LOADER to search for a new or next library record on paper tape.

(5) Depressing START after all subroutines are loaded causes a branch to the first instruction of the main program just loaded.

4.3.2 EOJ - Typeout

When an EOJ ($) block is encountered, the loader types out EJ. If no external subroutines are requested by the main program, it halts. When an EOJ ($) code is encountered at the end of a library input file, the loader does the following according to the settings of sense switches numbers 1 and 2:

(1) Both neutral: No subroutine name is listed.

(2) Number 1 raised, number 2 neutral: All subroutine names are listed.

(3) Number 1 neutral, number 2 raised: Only unloaded subroutine names are listed.

(4) Both on: All subroutine names are listed.

When the start button is depressed the loader either transfers control to the start location of the loaded program (if the number of undefined subroutine calls remaining is 0), or reads more information from the next library file.

4.4 RECOVERY PROCEDURES

4.4.1 "CK" Checksum Error

(1) Each block of input is concluded with a logical difference checksum. An error in the checksum causes the LOADER to typeout "CK" with a program HALT.

(2) Loading resumes with depression of the START toggle.

4.4.2 "MO" Memory Overflow

(1) "MO" will be typed followed by a program HALT when the program exceeds the available memory space, or if there are more literals and intermap references than can fit in one map.

(2) Depressing the START toggle results in a memory map typeout.

4.4.3 ABSOLUTE LOADER

Sense switch 0 SET - Load from High Speed Reader
RESET - Load from ASR-33 Reader

Sense switch 1 SET - Load Intermap references after loading program.

Starting location of absolute loader is normally memory high - 104_8.

ERROR MESSAGE

If a checksum error is found while loading tape with absolute loader, a message "K" is typed on ASR-33.

4.5 UPDATE/DEBUG UTILITY PROCEDURES

4.5.1 Update Instructions and Procedures

The UPDATE program (Catalog No. 300011A) is designed to allow the operator to easily correct or modify a symbolic source program by providing the following functions:

(1) Deletion of a specified line or group of lines.

(2) Insertion of a new or replacement line or lines.

(3) List the source program complete with line reference numbers.

All references to the symbolic source tape are made by referring to a sequence number. The sequence number is present on all assembly typeouts and on all typeouts generated by the UPDATE program.

Each function is initiated by a keyword entry on the Console Keyboard. The functions are all initiated by a type-in consisting of the SLASH CHARACTER (/), and the TERMINATOR (CARRIAGE RETURN key).

The keyword processor of the UPDATE program looks only at the first character, digits 0 through 9, and the terminator (Carriage Return). All other characters are ignored. The operator, therefore, can be as verbose or as brief as he desires.

Example:

/D20-29 CR	(Brief)
/DELETE ALL LINES BETWEEN 20 and 29 CR	(Verbose)
/I33 CR	(Brief)
/INSERT CORRECTIONS AFTER LINE 33 CR	(Verbose)

Sense Switch Settings

1 NEUTRAL	READ KEYWORDS FROM EXTERNAL DEVICE
1 RAISED	READ KEYWORDS FROM MEMORY
BOTH #2 and #3 NEUTRAL	KEYWORDS READ FROM CONSOLE KEYBOARD
2 NEUTRAL 3 RAISED	KEYWORDS READ FROM HIGH SPEED READER
2 RAISED 3 NEUTRAL	KEYWORDS READ FROM CONSOLE PAPER TAPE READER
4 NEUTRAL	SOURCE TAPE READ FROM HIGH SPEED PAPER TAPE READER
4 RAISED	SOURCE TAPE READ FROM CONSOLE PAPER TAPE READER
5 NEUTRAL	NEW SOURCE TAPE PUNCHED ON HIGH SPEED PAPER TAPE PUNCH
5 RAISED	NEW SOURCE TAPE PUNCHED ON CONSOLE PAPER TAPE PUNCH
6 NEUTRAL	NO LISTING DURING UPDATE PROCESS
6 RAISED	GENERATE CONSOLE PRINTER LISTING OF NEW SOURCE PROGRAM

NOTE

When No. 1 is RAISED, keywords (and corrections) are first read into the memory (starting at the end of the update program) from the device specified by No. 2 and No. 3, then, during updating, the keywords are retrieved from memory when needed. This procedure is useful when only one input device is available.

Keyword Definitions

1. Delete Keyword: /Daaaa CR
   or
   Keyword: /Daaaa bbbb CR

   Definition: Copy onto the new source tape from the old source tape all lines starting with the current line number to, but not including, line number aaaa.

   Then read line number aaaa (to and including line number bbbb if present) from the old source tape, but do not copy it onto the new source tape.

   After typing the last line deleted, call for a new keyword.

2. Insert Keyword: /Iaaaa CR

   Definition: Copy onto the new source tape from the old source tape all lines starting with the current line number to and including line number aaaa.

   After typing line number aaaa, call for a new keyword.

3. Data — Any line which does not start with a slash character is considered an insert line and will be punched verbatim onto the new source tape.

4. End Keyword: /E CR

   Definition: Terminate the new source tape by punching a form character (\) and then leader.

5. Done Keyword: /D CR

   Definition: Copy all lines from the current line on the old source tape to the last line on the old source tape onto the new source tape. Then terminate the new source tape.

Example Keyword List

```
Operator types:      /Delete 40 to 45  CR
ASR-33 Responds:     A  LAA  TOM  CR
                        AMA  JERY,1  CR
                        BRU  X + 4  CR
Operator types:      /D  50  CR
Operator types:      /I  68  CR
ASR-33 Responds:        AMA = 6  CR
Operator types:      /DONE  CR
```

Copy lines 1 to 39 from the old source tape to the new source tape. Skip lines 40 to 45, then punch the next three lines from the keyword list onto the new source tape. Now copy lines 46 to 49 onto the new source tape, skip over line 50 and copy line 51 to (and including) line number 68. After punching the next line in the keyword onto the new object tape, copy lines 69 to the end of the old source tape. Close out the tape and HALT.

If SENSE switch 1 is RAISED, all the keywords are read into memory from the device determined by SENSE switches 2 and 3. A HALT follows after the keywords are read. When the computer is restarted, the old source tape is read from the device specified by SENSE switch 4, processed according to the keyword list stored into memory, and the new source tape punched on the device specified by SENSE switch 5. If SENSE switch 1 was NEUTRAL the keywords are read from the specified device only when new keyword information is required. Two input devices are needed if SENSE switch 1 is NEUTRAL. Listing is controlled by SENSE switch 6. When it is RAISED, all information punched on the new object tape is also listed along with new line sequence numbers on the Keyboard Printer. When it is NEUTRAL, only the last line processed by any one keyword is typed (to indicate completion of the operation). In the example, lines 45, 50 and 68 would be typed.

4.5.2 Debug Instructions and Procedures

The DEBUG program (Catalog No. 300010) is a utility program designed to help a programmer debug a program while it is in memory. The following functions are provided:

(1) Type the contents of specified memory in octal or command format.

(2) Modify the specified memory. Input is in octal format or command format.

(3) Dump specified memory areas onto paper tape in a self-loading (non-relocatable) format.

(4) Load Binary tape.

(5) Enter breakpoints in order to "leap-frog" trace a program.

(6) Clear specified areas of memory to zero.

(7) Search memory for references to specified areas.

(8) Initiate branches (or HALT and BRANCH) to any part of memory.

Each of these functions is described in detail in later paragraphs and are initiated by typing a keyword through the Console Keyboard. This keyword consists of a letter, an address (or addresses) and the terminator, which is the Carriage Return key.

When a keyword requires two addresses (a lower and upper boundary), separate the addresses with a space or comma.

If an error is generated during input from the keyboard but before the keyword was terminated, type a slash character (/). This causes the keyword in error to be ignored, causes the Console Keyboard to generate a carriage return, and asks for a new keyword input. If the computer detects an error, it initiates the same action automatically.

The keyword input portion of the DEBUG program looks only at the first character, digits 0 through 7, and the terminators (carriage return). All other characters are ignored. The leading zero's on octal entries or addresses are not necessary.

Example:

Operator types: S20 CR
ASR-33 Responds: T140 153 CR

The control switches have meaning to the DEBUG program and are checked.

SENSE SW 0 - RAISED - HIGH SPEED PAPER TAPE LOAD/DUMP
NEUTRAL - ASR PAPER TAPE LOAD/DUMP

Type Memory Area-Octal

Keyword: Taaaaa bbbbb Terminator
Keyword: Taaaaa Terminator

Definition: Type in octal format, the memory words from location aaaaa to and including bbbbb. If bbbbb is less than aaaaa, or if only one address is given, only the word at location aaaaa is typed. A carriage return is typed after each set of four words.

Example:

Operator types: T1006 1011 CR
ASR-33 Responds: 01773245 14201017 42507762 31501732

Operator Types: T566 CR
ASR-33 Responds: 42477765

Type Accumulators

Keyword: R Terminator

Definition: Type the A and B Accumulators in octal format as shown in the example.

Example:

Operator types: R CR
ASR-33 Responds: 41217763 00721133

Type Memory Area-Command Format

Keyword: Caaaaa bbbbb CR
Keyword: Caaaaa CR

Definition: Type in symbolic command format, the memory words from location aaaaa to and including bbbbb. If bbbbb is less than aaaaa, or if only one address is given, only the instructions at aaaaa are typed. A carriage return is typed after each set of four words.

Examples:

Operator types: C2006 2010 CR
ASR-33 Responds: 12.101.307 04.000.711 07.001.434

Operator types: C56 CR
ASR-33 Responds: 10.010.336

Input Into Memory

Keyword: Iaaaaa Terminator

Definition: Set the address where the next octal or command format input word is to be stored. If a sequence of octal words is to be entered, aaaaa represents the starting address of that sequence.

Octal Input Data

Keyword: ±ddddd Terminator

Definition: The keyword is stored into memory at the location last specified by an Iaaaaa keyword. The address aaaaa is then incremented by 1. (When entering a sequential block of data, it is not necessary to precede each octal data with an Iaaaaa keyword, only the first word.) If six digits are not present, leading zeros are presumed.

Command Format Input Data

Keyword: OO. XIM. AAA Terminator

OO = Operator Code (00-17)
X = Index Bit (1 or 0)
I = Index Bit (1 or 0)
M = Map Bit (1 or 0)
AAA = Operand Address (000-777)

Definition: The keyword is condensed into binary form and stored into memory at the location last specified by an Iaaaaa keyword. The address aaaaa is then incremented by 1.

Examples:

Operator types: I2006 CR
ASR-33 Responds: I2, 101, 307 CR
ASR-33 Responds: 04, 000, 711 CR
ASR-33 Responds: 07, 001, 434 CR

Operator types: I00056 CR
ASR-33 Responds: 10, 010, 336 CR

Dump Memory on Paper Tape

Keyword: Daaaaa bbbbb Terminator

Definition: The memory area starting with aaaaa to and including bbbbb is punched on paper tape in a non-relocatable self-loading format.

Set Breakpoint

Keyword: Baaaaa Terminator

Definition: The contents of location aaaaa are saved and an SPB instruction is stored in its place. When this SPB instruction is executed, it causes the original instruction to be restored and a line of output to be typed on the console typewriter as follows:

aaaaa IIIIIII AAAAAAAA BBBBBBBB

aaaaa = location of instruction about to be executed.

IIIIIII = instruction about to be executed in command format.

AAAAAAAA = contents of A Accumulator before execution.

BBBBBBBB = contents of B Accumulator before execution.

After typing this line, the program cycles on waiting for a new keyword input.

Set Next Breakpoint

Keyword: N Terminator

Definition: The address aaaaa from the last Baaaaa keyword is incremented by 1 and used as the address for this keyword. After setting a Breakpoint (as described above) into this location, a transfer is made to that location-1. In this way, the operator traces each instruction in a sequential list of instructions with the minimum of effort.

NOTE

If the last instruction traced is a branch or skip type of instruction, this keyword cases the next sequential instruction to be traced only when it is actually executed.

Clear Memory

Keyword: Zaaaaa bbbbb Terminator
or
Keyword: Zaaaaa Terminator

Definition: Set to zero the memory locations starting at location aaaaa to and including bbbbb. If only one address is given, only that cell is set to zero.

Address Search

Keyword: Maaaaa bbbbb CR
or
Xxxx yyy CR
Keyword: Maaaaa bbbbb CR
or
Xxxx CR
Keyword: Xxxx yyy CR
or
Keyword: Xxxx CR

Definition Search memory from location aaaaa to and including bbbbb for any word having its address bits within the range of xxx to yyy inclusive. For each such word found, type the location of the word followed by the word itself in octal format.

If no M keyword is given, the memory area specified by the last M keyword is still in effect. If no yyy is specified, only addresses xxx are searched for.

Example:

Operator types: M1000 1777 CR

Operator types: X242 247 CR

ASR-33 Responds: 01227 001243
ASR-33 Responds: 01464 601247
ASR-33 Responds: 01470 401247
ASR-33 Responds: 01621 601242

Start Compute

Keyword: Saaaaa Terminator

Definition: When entering the DEBUG program either by an initial entry or by a Breakpoint entry, the contents of the registers are saved. The start keyword restores the registers to the state they were in at entry and causes a branch to location aaaaa.

Halt and Branch

Keyword: Haaaaa Terminator

Definition: This keyword is identical to the Saaaaa keyword except that after the registers are restored and just before branching, a halt takes place. This allows the operator to switch to the single step mode of computation.

Load Binary Tape L Terminator

Loads paper tape that was dumped by the D keyword.

Table 4-1 presents a summary of the above keywords.

Table 4-1. Keyword Summary

Taaaaa bbbbb	TYPE MEMORY AREA, OCTAL FORMAT.
Taaaaa	TYPE MEMORY WORD, OCTAL FORMAT.
R	TYPE ACCUMULATORS, OCTAL FORMAT.
Caaaaa bbbbb	TYPE MEMORY AREA, COMMAND FORMAT.
Caaaaa	TYPE MEMORY WORD, COMMAND FORMAT.
Iaaaaa	SET NEXT INPUT ADDRESS.
+ddddd	OCTAL INPUT DATA.
OO. XIM. AAA	COMMAND FORMAT INPUT DATA.
Daaaaa bbbbb	DUMP MEMORY AREA ONTO PAPER TAPE.
Baaaaa	SET BREAKPOINT.
N	SET BREAKPOINT INTO NEXT LOCATION.
Zaaaaa bbbbb	SET MEMORY AREA TO ZERO.
Zaaaaa	SET MEMORY WORD TO ZERO.
Maaaaa bbbbb CR Xxxx yyy	SEARCH SPECIFIED MEMORY AREA FOR SPECIFIED ADDRESS RANGE. *
Maaaaa bbbbb CR Xxxx	SEARCH SPECIFIED MEMORY FOR SPECIFIED ADDRESS. *
Xxxxx yyyy	SEARCH PREVIOUS MEMORY AREA FOR SPECIFIED ADDRESS RANGE. *
Xxxxx	SEARCH PREVIOUS MEMORY AREA FOR SPECIFIED ADDRESS. *
Saaaaa	START COMPUTE.
Haaaaa	HALT, OPERATOR OPTION.
L	LOAD BINARY TAPE.

*These are beginning and ending addresses.

SECTION 5

SOFTWARE MNEMBLER ASSEMBLER PACKAGE

5.1 INTRODUCTION

The SEL 810A MNEMBLER ASSEMBLER (Catalog No. 300009A) is a one or two pass symbolic assembly program which will accept symbolic instructions (source program) from a variety of input devices.

The output is an object program on either binary relocatable cards, paper tape, or magnetic tape ready for loading into the computer (object cards or tape). An optional symbolic listing with error messages and a side-by-side octal listing on the Console Keyboard Printer may also be selected for output.

5.1.1 Computer Configuration

A minimum of 4096 words of memory is required for the ASSEMBLER along with a Console Keyboard, Paper Tape Reader and Punch. No optional instructions are required.

The ASSEMBLER is easily modified to allow the use of larger memories or the use of the card reader, magnetic tape, and high-speed line printer when these devices are available.

5.1.2 Assembler Modes

The ASSEMBLER has two modes of operation, ONE PASS and TWO PASS, with the desired mode being determined by SENSE switch settings. For either mode, a paper tape, binary card, or magnetic tape is produced representing the assembled object program in a binary relocatable format acceptable to the Loader.

The ONE PASS mode of assembly is more desirable for use with the basic SEL 810A System because of the shorter time required for reading and punching tapes or cards. In the ONE PASS mode the source program is being assembled and a symbolic listing is generated complete with octal equivalence and any error messages.

The TWO PASS mode provides greater object program detail by providing a side-by-side octal listing of a completed assembly, including all error messages integrated with the assembly line effected. The output tape (or cards) is approximately 30 percent shorter than that produced by the ONE PASS mode of assembly but the processing time is considerably longer.

5.2 OPERATING PROCEDURES FOR MNEMBLER ASSEMBLER

5.2.1 Paper Tape Preparation of Source Program

Paper tape source programs are prepared using the Console Keyboard in the following manner:

Step 1 — Place the Console Keyboard in the OFF-LINE MODE.

Step 2 — Turn punch ON and depress the HERE IS key on the Console Keyboard (provides leader for the Paper Tape Punch).

Step 3 — Type in the instructions and data words from a MNEMBLER ASSEMBLER SOURCE PROGRAM coding form. Include comments if so desired.

Step 4 — Depress the CARRIAGE RETURN (CR) and LINE FEED after each statement line.

Step 5 — Complete the preparation of the source program.

Step 6 — The last two instructions on the MNEMBLER ASSEMBLER coding form are as follows:

LOC		OPER		ADDRESS, INDEX	
1		6		11	
			END		FOLLOWED BY CARRIAGE RETURN AND LINE FEED
$					FOLLOWED BY CARRIAGE RETURN AND LINE FEED

Step 7 — The last character on the paper tape should be a FORM character; this character is used by the UPDATE package as TERMINATOR.

5.2.2 Deletion of Errors on the Source Tape

Errors encountered during punching of the original source input tape are deleted if they are discovered before punching the LINE FEED and CARRIAGE RETURN for that statement line by depressing the UP ARROW. The ASSEMBLER ignores the statement line when the UP ARROW is depressed.

5.2.3 Loading the MNEMBLER ASSEMBLER

The SEL 810A MNEMBLER ASSEMBLER PAPER TAPE SOURCE PROGRAM is in relocatable format. The ASSEMBLER must be loaded with the relocatable Loader as described in Section 4 of this manual. If the Loader is not resident in memory prior to assembly, follow these initial procedures:

(1) Load into memory a binary non-relocatable paper tape of the Loader.

The operating procedures to load a binary non-relocatable paper tape, using either the MANUAL BOOTSTRAP or the BINARY PAPER TAPE READ PROGRAM, are found in Section 3.3 of this manual.

(2) The Assembler has an origin address of 00000; the operating procedures assume this origin. A new origin is entered into the program counter if the Assembler is relocated.

5.2.4 Assembling a Paper Tape Source Program (Typical)

Step 1 Rotate the Teletype On Line/Off/Off Line switch to the ON LINE position.

Step 2 Place the Paper Tape Reader START, STOP, FREE toggle switch in the START position.

Step 3 Place the paper tape source program in position on the Paper Tape Reader.

Step 4 Raise the appropriate SENSE switches on the Console Control Panel to designate mode of assembly and I/O devices to be used.

Step 5 Choose assembly mode --
a) ONE PASS MODE -- FOLLOW STEPS 6-7.
b) TWO PASS MODE-- FOLLOW STEPS 6-10.

Step 6 Ready the Console Paper Tape Punch by depressing the ON button located on the punch unit.

Step 7 Depress START switch on the Computer Control Panel.

NOTE

A ONE PASS or TWO PASS assembly is terminated when the ASSEMBLER reads the ($) punch on the paper tape source program. If a TWO PASS assembly is designated, the assembly will HALT the computer upon recognizing the END statement punched in the paper tape source program at the end of first pass.

Step 8 Reposition (rethread) the source program paper tape into the Paper Tape Reader to allow the Assembler to complete PASS TWO of the assembly.

Step 9 Depress the ON button on the Console Paper Tape Punch.

Step 10 Depress START toggle on the Computer Control Panel.

5.2.5 Selection of Assembler Options by SENSE Switch Settings

The options available for use by the Assembler are controlled by the SENSE SWITCH settings on the Computer Control Panel. The following sense switch settings enable the MNEMBLER ASSEMBLER to perform the desired mode of assembly and indicate the type of devices used for source program input and object program output. SENSE SWITCHES (Entry Toggles) are operated in the RAISED or NEUTRAL position. The following table defines options (functions) versus position for each SENSE SWITCH.

SENSE SWITCH	POSITION	FUNCTION
0	RAISED	TWO PASS ASSEMBLY
0	NEUTRAL	ONE PASS ASSEMBLY
1	RAISED	NO SYMBOLIC OUTPUT (NORMALLY LISTING)
1	NEUTRAL	SYMBOLIC OUTPUT
2	RAISED	NO OBJECT OUTPUT
2	NEUTRAL	OBJECT OUTPUT PROVIDED
3	RAISED	LIST THE ERROR LINES ONLY
3	NEUTRAL	LIST ACCORDING TO SENSE SWITCH 1
4	RAISED	LISTING ON CONSOLE KEYBOARD PRINTER
4	NEUTRAL	LISTING ON HI-SPEED (OPTIONAL) PRINTER
5	RAISED	SOURCE INPUT FROM PAPER TAPE READER
5	NEUTRAL	SOURCE INPUT FROM (UPDATE) CARD READER
6	RAISED	OBJECT OUTPUT ON PAPER TAPE PUNCH
6	NEUTRAL	OBJECT OUTPUT ON HI-SPEED (OPTIONAL) PAPER TAPE PUNCH IF S.SW 12 RESET
7	RAISED	LIST THE SYMBOL TABLE
7	NEUTRAL	SYMBOL TABLE NOT LISTED
8	RAISED	SOURCE INPUT FROM ASR 33 PAPER TAPE READER
8	NEUTRAL	SOURCE INPUT FROM HI-SPEED PAPER TAPE READER
9	RAISED	SOURCE INPUT FROM CONSOLE KEYBOARD
9	NEUTRAL	SELECTS INPUT ACCORDING TO SENSE SWITCHES 5 AND 8.
*10	RAISED	START PASS TWO AFTER PASS TWO
*10	NEUTRAL	START PASS ONE AFTER PASS TWO
11	RAISED	MAG TAPE SOURCE INPUT

Continued on next page.

SENSE SWITCH	POSITION	FUNCTION
12	RAISED	MAG TAPE OBJECT OUTPUT
13	RAISED	MAG TAPE SYMBOLIC OUTPUT

*The following procedure must be followed to obtain a listing and/or object paper tape output when the Console Keyboard Printer is used to output the object paper tape during assembly:

1) ONE PASS MODE -- Process the source input from any device other than the Console Keyboard with SENSE SWITCHES 1 and 6 raised to produce an object paper tape output. If an assembly listing is then required, raise SENSE SWITCH 2 and depress the START toggle to produce an assembly listing.

2) TWO PASS MODE -- Process the source input from any device other than the Console Keyboard with SENSE SWITCHES 0, 1, and 6 raised on the second pass to produce output on paper tape.

   If an assembly listing is required, set SENSE SWITCH 1 to its NEUTRAL position and set SENSE SWITCHES 2 and 10 to the RAISED position.

   Depress the START toggle to produce an assembly listing.

3) SENSE SWITCHES 1, 2, 3, in the RAISED state produce no output other than a listing of the source program statement line in error.

5.2.6 Symbolic Listing Format

The symbolic listing is optional and when requested, is made on the Console Keyboard Printer (device determined by sense switch setting). A typical line has the following format:

```
016  01733M01301745  ALPH  LAA*DATA     TEST FLAG
017  01734  00000610        RSA 6        NEXT TEST
018  01735  11101360        BRU AL30     RESTART
019  01736  00000001        DATA 1, 2, 3 SWITCHES
020  01737  00000002
021  01740  00000003
022  01741  00000004  LIST  BSS  4        LIST
023  01745  00101733  DATA  DAC  ALPH, 1  COUPLING
```

(5) Card/Line Image (80 if line printer; 54 if teletype)

(4) Side-by-Side Octal Instructions (8 Octal Digits)

(3) Error Flag (1 Letter)

(2) Instruction's Memory Location (5 Octal Digits)

(1) Line Number for update purposes (3 Decimal Digits)

(1) This decimal number is referred to when updating a source tape using the Update Program explained in Section 5.4.

(2) This octal number represents either the actual or relative memory location where the assembled instruction is stored.

(3) This letter represents a suspected error situation within the line. (See Section 5.3).

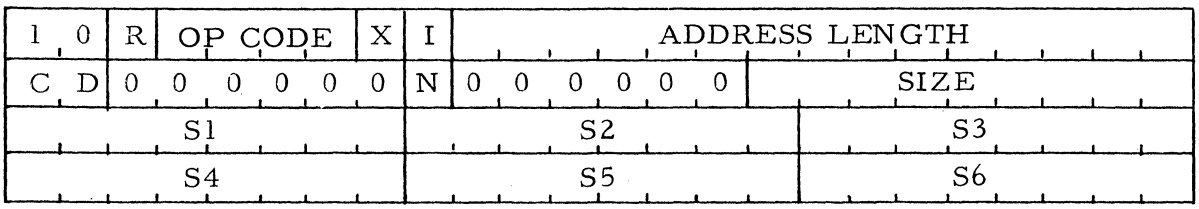
(4) The octal listing has one of five formats. The five formats are:

(1) o o c a a a a a — Assembled instruction usually resulting from a DATA, DAC or EQU pseudo-operation instruction where c a a a a a is any octal number with c equal to one or zero. The c a a a a a portion of the listing is loaded into memory exactly as it appears.

(2) o o o o b b x x — Assembled augment instruction where b b is up to 6 augmenting bits and x x is the operation code. The instruction appears in memory as o o b b x x.

(3) x x y z z z z z — Assembled memory referencing instruction where x x is the operation code, y is index, indirect and relocation bits. z z z z z is the relative address. The x x portion of the listed instruction is the only part which <u>always</u> appears the same in memory. The other portions are altered depending on how the loader loads them.

(4) Subroutine or Common:



CD = 10: Common definition
Address = length

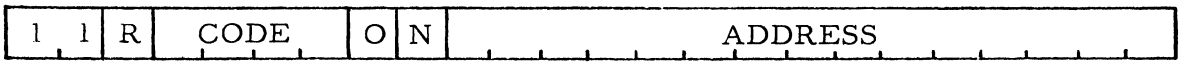
CD = 11: Common request
address = relative to block
N = negative flag

CD = 00: Subroutine defintion (NAME)
address = relative entry point

CD = 01: Subroutine call (CALL)
address = 0

This format appears on the listing as well as the object output and is information to the loader only. It generates an SPB and loads it into memory for a subroutine call.

(5) This format will appear on the listing as well as the object output and is information to the loader only.



Code = 00, Establish Load Point (ORG)
= 01, END Jump (END)
= 02, STRING
= 03, 9-bit ADD-TO (from one-pass assembly)
= 04, 14-bit ADD-TO (DAC) (from one-pass assembly)
= 05, 15-bit ADD-TO (EAC) (from one-pass assembly)
= 06, Turn on CHAIN flag
= 07, Turn on Load flag
= 10, END-OF-JOB ($)

(5) The complete input line is listed in its original format, if output is on the line printer, or the first 50 characters of the input line are output if output is on the Console Keyboard Printer.

5.3 OPERATOR COMMUNICATIONS

Error flag symbols given in the Assembler listing have the following meaning:

SYMBOL	MEANING
U	Undefined Symbol
Q	Undefined Operation Name
M	Multiple Defined Symbol
A	Address Field Missing
S	Use of Map In One Pass Mode or with Program Exceeding 512 Words of Memory
D	Data Conversion Exceeds Limits
T	Assignment Table Full
E	Any Other Type of Detected Error

NOTE:

Errors will cause the affected fields to be set to zero. More than one error may be detected for each line of coding, but only the last error is flagged in the assembly listing.

5.4 RECOVERY PROCEDURES

5.4.1 Table Size

The assignment table contains all symbols defined in the source program being assembled, plus all literal constants, subroutine names and modifier instructions. On a 4096 word memory computer configuration, the number of entries is approximately determined as follows:

$$600 = S+L+C+M$$

where

S = number of symbolic addresses

L = number of literal constants defined

C = number of subroutine names called

M = number of times an "undefined" symbol appears in the variable field of an instruction combined with constants by a (+) or (-).

If the computer configuration is greater than 4096 words of memory, the assignment table is easily expanded and the following equation holds:

$$600 + 1365 * (N-1) = S+L+C+M$$

where

N = number of 4096 memory modules available.

5.5 UPDATE/DEBUG UTILITY PROCEDURES

5.5.1 Update Instructions and Procedures

The UPDATE program is designed to allow the operator to easily correct or modify a symbolic source program by providing the following functions:

(1) Deletion of a specified line or a group of lines.

(2) Insertion of a new or replacement line or lines.

(3) List the source program complete with line reference numbers.

All references to the symbolic source tape are made by referring to a sequence number. The sequence number is present on all assembly typeouts generated by the UPDATE program.

Each function is initiated by a keyword entry on the Console Keyboard Printer. The functions are all initiated by a type-in consisting of the SLASH CHARACTER (/), and TERMINATOR (CARRIAGE RETURN).

The keyword processor of the UPDATE program looks only at the first character, digits 0 through 9, and the terminators (carriage return). All other characters are ignored. The operator, therefore, can be as verbose or as brief as he desires.

Example:

/D20-29 CR	(Brief)
/DELETE ALL LINES BETWEEN 20 and 29 CR	(Verbose)
/I33 CR	(Brief)
/INSERT CORRECTIONS AFTER LINE 33 CR	(Verbose)

5.5.2 Sense Switch Settings

1 NEUTRAL	READ KEYWORDS FROM EXTERNAL DEVICE
1 RAISED	READ KEYWORDS FROM MEMORY
2 NEUTRAL 3 NEUTRAL	KEYWORDS READ FROM CONSOLE KEYBOARD
2 RAISED 3 NEUTRAL	KEYWORDS READ FROM CONSOLE PAPER TAPE READER
4 NEUTRAL	SOURCE TAPE READ FROM HIGH SPEED PAPER TAPE READER
4 RAISED	SOURCE TAPE READ FROM CONSOLE PAPER TAPE READER
5 RAISED	NEW SOURCE TAPE PUNCHED ON HIGH SPEED PAPER TAPE PUNCH
5 NEUTRAL	NEW SOURCE TAPE PUNCHED ON CONSOLE PAPER TAPE PUNCH
6 NEUTRAL	NO LISTING DURING UPDATE PROCESS
6 RAISED	GENERATE CONSOLE PRINTER LISTING OF NEW SOURCE PROGRAM

NOTE

When No. 1 is RAISED, the keywords (and corrections) are first read into the memory (starting at the end of the update program) from the device specified by No. 2 and No. 3, then, during updating, the keywords are retrieved from memory when needed. This procedure is useful when only one input device is available.

NOTE

Detailed procedures are discussed in Section 4.5 of this manual.

Listing is controlled by SENSE switch No. 6. When it is RAISED, all information punched on the new object tape is also listed along with new line sequence numbers on the Keyboard Printer. When it is NEUTRAL, only the last line processed by any one keyword is typed (to indicate completion of the operation). In the Example Keyword List, Section 4.5.1, lines 45, 50 and 68 would be typed.

5.5.3 Debug Instructions and Procedures

The DEBUG program is a utility program designed to help a programmer debug a program while it is in memory. The following functions are provided:

(1) Type the contents of specified memory in octal or command format.

(2) Modify the specified memory. Input is in octal format or command format.

(3) Dump specified memory areas into paper tape in a self-loading (non-relocatable) format.

(4) Load binary tape.

(5) Enter breakpoints in order to "leap-frog" trace a program.

(6) Clear specified areas of memory to zero.

(7) Search memory for references to specified areas.

(8) Initiate branches (or HALT and BRANCH) to any part of memory.

Each of these functions are described in detail in later paragraphs and are initiated by typing a keyword through the Console Keyboard. This keyword consists of a letter, an address (or addresses) and a terminator.

When a keyword requires two addresses (a lower and upper boundary), separate the addresses with a space or comma.

If an error is generated during input of the keyboard but before the keyword was terminated, type a slash character (/). This causes the keyword in error to be completely ignored, the Console Keyboard to generate a carriage return, and asks for a new keyword input. If the computer detects an error, it will initiate the same action automatically.

The keyword input portion of the DEBUG program looks only at the first character, digits 0 through 7, and the terminator (carriage return). All other characters are ignored. The leading zeros on octal entries or addresses are not necessary. The operator, therefore, can be as verbose or as brief as he desires.

Example:

Operator Types: S20 CR
ASR-33 Responds: T140-153

The control switches have no meaning to the DEBUG program and are never checked.

NOTE

Detailed procedures are discussed in Section 4.5.

SECTION 6

SOFTWARE FORTRAN IV COMPILER PACKAGE

6.1 INTRODUCTION

The FORTRAN IV COMPILER (Catalog No. 344001A) operates in a minimum configuration of 8192 words of memory with an input/output consisting of the console keyboard printer.

All input/output is performed using logical device unit numbers. These unit numbers are transmitted to the compiler via an initial setting of the A Accumulator.

Other options are transmitted via the A Accumulator, such as Binary Output Control, Source and Symbolic Output, Ignore Trace, and Compile Library Option. Figure 6-1 specifies the meaning of the initial settings for the desired options and input/output devices.

The FORTRAN COMPILER compiles one program or a series of programs. An END-OF-JOB is indicated by a "$" character in column 1 of the punched card following the last END card.

During compilation, source errors are detected and output with the statement in error appropriately underlined. This consists of the comment ".....ERROR" positioned at the place the compiler examines after the error is detected.

On entrance to the FORTRAN COMPILER, the A Accumulator is interrogated to determine the particular options selected for this compilation. (Refer to Figure 6-1).

6.2 OPERATING PROCEDURES

Step 1 — Using the SOFTWARE LOADER PACKAGE, load the FORTRAN compiler Package. (Refer to Section 4.)

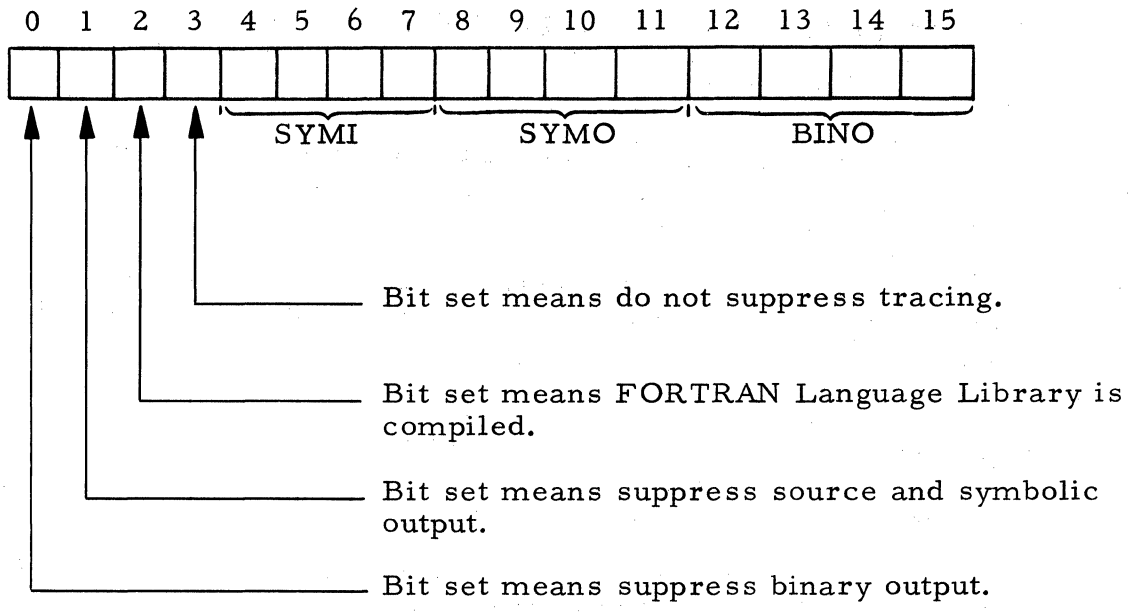
Step 2 — Set the A Accumulator for the desired options and input/output devices. (Refer to Figure 2-5).

Step 3 — Set the PROGRAM COUNTER to address 01000.

Step 4 — Place the source program in the selected input device.

Step 5 — Depress the START toggle switch on the Computer Control Panel.

A ACCUMULATOR



Where SYMI is the symbolic input unit number.
SYMO is the symbolic output unit number.
BINO is the binary output unit number.

INPUT UNIT NUMBERS

SYMI = 0001 --- Selects the Console Keyboard.
= 0010 --- Selects the Hi-Speed Paper Tape Reader.
= 0011 --- Selects the Card Reader.
= 0100 --- Not Assigned.
= 0101 --- Selects the Console Keyboard Paper Tape Reader.

OUTPUT UNIT NUMBERS

SYMO = 0001 --- Selects the Console Keyboard Printer.
= 0010 --- Selects the Hi-Speed Paper Tape Punch.
= 0011 --- Selects the Card Punch.
= 0100 --- Selects the Hi-Speed Line Printer.
= 0101 --- Selects the Console Keyboard Paper Tape Punch.

BINARY OUTPUT UNIT NUMBERS

BINO = 0001 --- Selects the Console Keyboard Printer.
= 0010 --- Selects the Hi-Speed Paper Tape Punch.
= 0011 --- Selects the Card Punch.
= 0100 --- Selects the Hi-Speed Line Printer.
= 0101 --- Selects the Console Keyboard Paper Tape Punch.

Figure 6-1.

6.2.1 Executing Compiler Generated Programs

Step 1 — Load the software LOADER (Refer to Section 4).

Step 2 — Set the A Accumulator to the starting location address for the program being loaded. Set to zero if relocation is not required.

Step 3 — Set the B Accumulator to the starting location address for the inter-map reference table. The B Accumulator must be greater than 10 if any library routines are used by the program being loaded.

Step 4 — Set the SENSE SWITCHES for LOADER options.

Step 5 — Place the compiler generated object program into the selected input device.

Step 6 — Depress the START toggle on the Computer Control Panel.

NOTE

It is necessary to load library tape after loading the relocatable object tape of the program to be compiled. Thus after loading complied tape it types LC and if no subroutines LCEJ.

"LCEJ"

After all the required library routines have been loaded, the LOADER will type "LCEJ", indicating that loading is complete.

Step 7 — Depress the START toggle switch on the Computer Control Panel to execute the loaded program.

6.2.2 Chaining for FORTRAN IV Programs

The Chain Program used with the SEL 810A LOADER provides a means for executing programs which exceed memory capacity. Chaining allows the operator to section a large program into smaller independent segments. Each segment (link) is compiled as an executable program unit (i.e., main program and subroutines as required).

Intermediate results, etc., are communicated between links during the chain execution by storing these results in the common region, provided each link is given identical common declarations (order and size).

Control is transferred from link-to-link by means of the FORTRAN statement CALL CHAIN which is the last executable statement of each link. Run-time execution of this statement within a link will initiate loading and execution of the next link.

6.2.3 Preparing a Chain Tape

Chain jobs are executed from a special "chain tape" prepared with the aid of the LOADER.

Step 1 — Following initial loading, load the first link as a normal program unit.

Step 2 — Following initial punchout, re-enter the LOADER at 77777_8. This causes the first segment of the chain tape to be punched.

Step 3 — Repeat steps 1 and 2 for the remaining links in order of execution. The chain tape is now complete.

6.2.4 Executing a Chain Tape

Step 1 — Enter the absolute loader at location 0.

Step 2 — Place the chain tape in the paper tape reader with the first character at the read head.

Step 3 — Start execution at location 0.

When the first segment transfers to the CALL CHAIN subroutine, this subroutine causes the next segment to be read from the special binary paper tape and that segment is executed. This operation is automatic and no operator action is required. Step 3 is repeated automatically until the last segment is loaded and executed.

Chain Program Example:

```
C     LINK NO. 1
      COMMON A, B
      WRITE (1, 1)
1     FORMAT (15H THIS IS LINK 1)
      A = 2*B
      CALL CHAIN
      END
```

Chain Program Example Cont'd:

```
$

C     LINK NO. 2
      COMMON X, Y
      WRITE (1, 1)
1     FORMAT (15H THIS IS LINK 2)
      Y = 2*X
      CALL CHAIN
      END

$

C     LINK NO. 3
      COMMON E, F
      WRITE (1, 1)
1     FORMAT (15H THIS IS LINK 3)
      E = 2*F
      CALL EXIT
      END

$
```

6.2.5 Trace

The trace facility of the SEL 810A FORTRAN system provides a powerful debugging aid allowing either selective tracing of specified variable/array values or program segments during program execution.

Tracing is specified by the FORTRAN source statement TRACE. Use of TRACE as the first executable program statement, followed by a list of variable and array names, will cause a coupling to run-time trace routine to be inserted into the object coding following every definition of (store into) the name. Typical TRACE output is illustrated in Figure 6-2.

Occurrence of the executable statement TRACE n ("n" statement number) causes the compiler to generate trace coupling after every variable, array element and IF expression subsequently defined until statement "n" is processed. Also, all numbered statements within this area cause the statement number to be output by the trace before being executed. This gives the programmer a logical flow path as part of his diagnostic information. As many of these TRACE statements as desired are strategically placed throughout the program. The tracing of the entire program is accomplished by the statement TRACE 99999.

6.2.6 Trace Output

At run-time trace output is controlled by the setting of SENSE SWITCH 4. If SENSE SWITCH 4 is raised, all trace output is inhibited. Otherwise, tracing proceeds as specified.

Several output formats result in the trace listing depending on the mode of the output (Figure 6-2). Each item being traced, whenever encountered, causes a line to be typed consisting of a variable name, an array name, or a $ "n", followed by an equal (=) sign, followed by the current decimal value just assigned to that name. The decimal value is typed in integer format, floating point format, or complex format. Array names are followed by a subscript indicating the element within the array just modified as if it were a single dimensioned array.

INTEG	=	17328	Integer Variable
REAL	=	-.32171964000E 02	Real Variable
LOGY	=	T	Logical Variable
ARRAY (14)	=	-.69133251902E-01	Array Element
(IF)	=	.19347782170E-01	If Expression Result
($10)			Statement Number
DBL	=	-.32171964000D 02	Double-precision Variable
CMPLX	=	(.9171978E 03,.1037200E 00)	Complex Variable

Figure 6-2. TRACE Listing Output Format

6.3 OPERATOR COMMUNICATIONS

FORTRAN IV Diagnostics

FORTRAN IV Diagnostic messages consist of the statement ERR typed on the line following the FORTRAN IV statement in error.

The following list contains the different diagnostic codes and their meaning.

CODE		ROUTINE	MEANING
ADDR			ILLEGAL ADDRESS CONSTRUCTION
ADJD		SC01	ILLEGAL ADJUSTABLE DIMENSIONS
AMOD			ILLEGAL MODE FOR ADDRESS (MUST BE INTEGER)
ASOV	*	AS03	ASSIGNMENT TABLE OVERFLOW
ASTO		X301	ASSIGN TO SPELLING ERROR
BLKD		W500	NO CODE GENERATED BY A BLOCK DATA PROGRAM
CERR	*	A100	CHARACTER NO A C/R

CODE		ROUTINE	MEANING
CICD			CANNOT INITIALIZE COMMON DATA
COMM		NM01	ERRONEOUS COMMON USAGE
CRET	*	TH02	C/R WITHIN HOLLERITH STRING
DDST		NR01	DOUBLY DEFINED STATEMENT
DPFL	*	B500	DATA POOL FULL
DPOF			DATA POOL OVERFLOW
DUMM		ND01	ERRONEOUS DUMMY USAGE
EQCN		C310	ERRONEOUS EQUIVALENCE CONSTRUCTION
EQIV		C309	IMPOSSIBLE EQUIVALENCE GROUP
EQMS	*	C901	DO EQUALS (=) IS MISSING
ERDO	*	C900	ILLEGAL DO-TYPE STATEMENT
ERTN	*	R903	RETURN STATEMENT IN MAIN PROGRAM
EXS=	*	EX66	NOT FIRST EQUALS, OR EQUALS WITH PARENTHESIS, OR EQUALS NOT ALLOWED
FUNV		W501	FUNCTION NAME NEVER ASSIGNED
FRST	*	R205	NOT FIRST STATEMENT OF PROGRAM
FWAR	*	R203	FUNCTION HAS NO ARGUMENTS
HOLL			ILLEGAL HOLLERITH STRING
IDOL	*	V516	IMPROPER IMPLIED DO LOOP
IF(2	*	V307	IF (ITEM HAS OVER 6 CHARACTERS
ILBD	*	R301	ILLEGAL BLOCK DATA STATEMENT
ILEG	*	A900	NOT LEGAL FORTRAN STATEMENT
ILIF			ILLEGAL LOGICAL IF CONSTRUCTION
ILSN	*	IS04	ILLEGAL STATEMENT NUMBER
INDT			DATA CONSTRUCTION ERROR
IUSE		NU00	INCORRECT USAGE
LDOP	*	EX79	IMPROPER LEADING OPERATOR
MODE	*	OMZ5	MODE MIXING ERROR
MULT		NP02	MULTIPLE DEFINED ITEM
NAME			CONSTANT ILLEGALLY USED
NARR		AT00	ITEM NOT AN ARRAY
NCBS		C315	NEGATIVE COMMON BASE
NEXT	*	C604	IMPROPER DO NEST
NINT		IT00	ITEM NOT AN INTEGER

CODE		ROUTINE	MEANING
NNAM		NCOO	ILLEGAL USE OF CONSTANT
NOIM			OPERAND MISSING
NOIT			MUST HAVE INTEGER TYPE
NPTH		V219	NO FORMAT STATEMENT NUMBER
NOC			ILLEGAL USE OF SUBROUTINE OR ARRAY NAME
OPER	*	EX25	UNACCEPTABLE OPERATOR
OPOS	*	EX60	OPERATOR NOT ALLOWED AT THIS POSITION
PATH		NP06	PATH CANNOT EXECUTE THIS STATEMENT
RLOP	*	EX70	TWO RELATIONAL OPERATORS IN A ROW
SBIG	*	DN57	DIGIT STRING TOO LARGE
SBSC		IL01	WRONG NUMBER OF SUBSCRIPTS
SPEC	*	NP00	STATEMENT CLASS OUT OF ORDER
SPEL		A903	FORTRAN STATEMENT MISSPELLED
STNO	*	C702	STATEMENT NO. CONSTRUCTION
TAG			ILLEGAL INDEX CONSTRUCTION
TYPE	*	A304	IMPROPER USE OF TYPE STATEMENT
TMDT			TOO MUCH DATA
V/SP		NS01	ILLEGAL USE OF SUBPROGRAM NAME
XARG			EXCESSIVE NUMBER OF ARGUMENTS
)ERR	*	TS01	CHARACTER NOT A)
(ERR	*	TS02	CHARACTER NOT A (
/ERR	*	TS03	CHARACTER NOT A /
,ERR	*	TS04	CHARACTER NOT A ,

* IRRECOVERABLE ERROR. ENTIRE RECORD IS IGNORED.

SECTION 7

HARDWARE DIAGNOSTICS

7.1 INTRODUCTION

The SEL 810A Diagnostic Program is a complete package designed to give the operator the ability to exercise the memory and the associated input/output peripheral equipment.

The memory exerciser routine generates various types of worst case bit patterns and exercises the memory with these patterns while monitoring for errors. Provisions are made for automatic relocating of the exerciser program to allow the entire memory to be included in all tests. Also, included are certain branch/skip instructions which are sequenced and executed through each location in the memory.

The mainframe exerciser routine executes the entire instruction repertoire individually in a large variety of sequences while monitoring the results for errors. Errors are indicated by a program halt. Pertinent information concerning the instruction that failed and the nature of the failure are obtained from the A and B Accumulator displays, the Program Counter and certain selected memory locations.

The programs for the associated I/O peripheral equipment tests the ability of the various I/O units to generate or receive all acceptable characters. A selected input is used and visual monitoring of the control panel or output unit is required by the operator for verification of proper operation. Equipment tested includes standard Teletype output, input, punch and reader as well as optional card punch, line printer, high-speed paper tape equipment, magnetic tape units and other units as needed for a particular application.

The following sections include the operating procedures, descriptions, error messages, and recovery actions for the hardware checkout programs. These procedures assume that the user is familiar with the operating instructions discussed in previous sections of this manual.

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

 Catalog No. 303009A

7.2 810A Mainframe Diagnostic Loading Procedures

AUTHOR: Systems Engineering Laboratories

ACCEPTED: February 23, 1967

PURPOSE: To provide load operating procedures for the mainframe diagnostic programs.

COMPUTER CONFIGURATION: SEL 810A with console typewriter or high speed paper tape reader.

SUBROUTINES REQUIRED: N/A

STORAGE: N/A

TIMING: N/A

USE:

A. Load the bootstrap shown in Figure 7-1 for ASR-33 input or Figure 7-2 for high speed reader.
B. Place the Standard Load-Dump program (Catalog No. 300001A) on either the ASR-33 or the high speed reader.
C. Set the proper sense and/or control switches on the console
   Sense switch zero Set: Load from high speed reader
   Reset: Load from ASR-33 reader
D. Raise the single cycle/no program advance switch on the console.
E. Press START on the console.
F. The Absolute Load program operating procedures (Catalog No. 300001A) will apply to the loading of the diagnostic programs once the Loader has been loaded into memory.

METHOD: A. Each mainframe diagnostic program must be loaded by using the Standard Absolute Load program (Catalog No. 300001A) or the Loader contained within debug.

Reference should be made to the Standard Absolute Load program for correct operating procedures and for core memory allocation of the load program.

When a program as been successfully loaded into memory, refer to the operating procedures associated with the diagnostic program.

B. The following diagnostic program, in absolute format, must be loaded by the Absolute Loader.

1.	Mainframe Exerciser	Catalog No. 303001A
2.	Instruction Simulation	Catalog No. 303002A
3.	CMASAS	Catalog No. 303003A
4.	MEMDEX	Catalog No. 303004A
5.	LSRCT	Catalog No. 303005A
6.	ADDØ	Catalog No. 303006A
7.	MTPY	Catalog No. 303007A
8.	Divide Test	Catalog No. 303008A
10.	MEMTES	Catalog No. 303010A

C. Memory Test Diagnostics

After the execution of the MEMDEX, MEMTES and CMASAS, the Absolute Loader must be loaded into memory since the three mentioned programs exercise memory.

ASSEMBLY CODING FORM

SEL

PROGRAMMER: Systems Engineering Laboratories

PAGE OF

PROGRAM: ASR-33

DATE

IDENTIFICATION 73 80

LOC. 1	OPER. 6	ADDRESS, INDEX 11	25 LOCATION	50 72
STRT	CEU	1,2,W	0	130101
	DATA	'4000	1	004000
	AIP	1,2,W	2	170301
	SAZ		3	000022
	BRU	*+2	4	111006
	BRU	*-3	5	111002
READ	AIP	1,2,W	6	170301
	LSL	8	7	001016
	AIP	1,2,W,R	10	174301
	STA*	DAC1	11	033016
	SAZ		12	000022
	IBS		13	000026
	BRU	DAC2	14	113017
	BRU	READ	15	111006
DAC1	DAC	CHAN-2,1	16	*
DAC2	DAC	CHAN	17	**
* 8K		117671	16K 137671	
** 8K		017673	16K 037673	

Figure 7-1

ASSEMBLY CODING FORM

SEL

PROGRAMMER: Systems Engineering Laboratories

PAGE OF

PROGRAM: High Speed Reader

DATE

IDENTIFICATION (73–80)

LOC. (1)	OPER. (6)	ADDRESS, INDEX (11)	LOCATION (25)	(50–72)
STRT	CEU	2,W	0	130102
	DATA	'1,000	1	001000
	AIP	2,W	2	170302
	SAZ		3	000022
	BRU	*+2	4	111006
READ	BRU	*-3	5	111002
	AIP	2,W	6	170302
	LSL	8	7	001016
	AIP	2,W,R	10	174301
	STA*	DAC	11	033016
	SAZ		12	000022
	IBS		13	000026
	BRU	DAC2	14	113017
	BRU	READ	15	111006
DAC1	DAC	CHAN-2,1	16	*
DAC2	DAC	CHAN	17	**
* 8K		117671	16K 137671	
** 8K		017673	16K 037673	

Figure 7-2

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

7.3 Mainframe Exerciser (MFE)

AUTHOR: SEL

ACCEPTED: 13 January 1967

PURPOSE: A fast no-loop program designed to use each of the mainframe instructions in such a way that if a Halt occurs, the operator can tell from the program listing which instruction is malfunctioning.

SOURCE PROGRAM LANGUAGE: MNEMBLER - 810A

COMPUTER CONFIGURATION: Standard SEL 810A

STORAGE: 2000_8 to 2303_8 3000_8 to 3021_8 - Not Relocatable

SUBROUTINES REQUIRED: 810A Mainframe Diagnostic Loading Procedure

TIMING Approx. 1 ms/cycle

USE: Start at location 2000_8, the program will run until halted manually. If a halt occurs consult the listing or halt log using the P-Counter location to find the instruction that failed.

HALT LOG FOR MAINFRAME EXERCISER

P-Counter Location	Instruction in Error
2006_8	AMA, SOF
2011_8	AMA
2020_8	AMB SOF
2023_8	AMB
2030_8	LSL
2034_8	RSA, SMA
2041_8	FLL

2044_8	FLL
2051_8	FRA
2053_8	FRA
2061_8	LSA
2065_8	LSA
2071_8	CLA
2102_8	FRL
2106_8, 2107_8	CMA
2112_8, 2114_8	CMA
2117_8, 2120_8	CMA
2125_8, 2127_8	STB
2134_8	ABA
2141_8	OBA
2153_8	NEG, CNS. SMA
2157_8, 2160_8	SAS
2163_8, 2165_8	SAS
2170_8, 2171_8	SAS
2200_8	FLA, SMA
2216_8	RSL. FRA, RNA, TAB, CLA, IAB, ASC
2222_8	SNO
2225_8	SNO
2232_8	LOB
2236_8	SMA
2241_8	IBS
2251_8, 2254_8	MPY
2262_8	MPY, SMA
2267_8	AMA, SMA, TBA
3003_8	BRU Indirect
3013_8	SOF
3016_8	SOF

METHOD: N/A

7.4	Instruction Simulation and Comparison (IS&C)
AUTHOR:	SEL
ACCEPTED:	13 January 1967
PURPOSE:	Executes mainframe instructions and simulates them if possible by software. The results are compared and an error condition occurs on an error. Some instructions cannot be simulated easily so they are executed and the results compared to a constant.
SOURCE PROGRAM LANGUAGE:	MNEMBLER 810A
COMPUTER CONFIGURATIONS:	Standard SEL 810A
STORAGE:	1000_8 to 2605_8, 0 to 22_8 - Not Relocatable
SUBROUTINES REQUIRED:	810A Mainframe Diagnostic Loading Procedure
TIMING:	Approx. 40 seconds/cycle, without errors
USE:	Start at location 1000_8. If an error occurs, consult the routine description to find what instruction failed. The program will run until halted manually.

Sense Switches

SSW 0 up - A successful cycle type-out will occur approximately every 40 seconds only if there have been no errors during that cycle.

SSW 1 up - Errors will be ignored.

SSW 2 up - No error type-out will occur, the machine will halt and the A Accumulator may be displayed for the error location.

SSW 3 up - A halt will occur after the error type-out.

TYPE-OUT FORMATS:

Successful Cycles - NNNN
NNNN = Decimal number of cycles in which no errors occurred.

Machine error preceeding location XXXXX
XXXXX = Octal location from which a SPB occurred after an error condition was found by the program.

METHOD:

Clear A Accumulator (CLAT)

The A Accumulator is loaded with the counter and cleared. A is then checked for zero. The counter is then incremented.

The test is repeated for every case. An error at location 1004_8 indicates a CLA error.

Skip if A Accumulator is Zero (SAZT)

The B Accumulator is incremented and transferred to A. A is checked for zero by the SAZ and then A is checked for zero by the CMA. An error will occur at location 1022_8 if a skip occurs when A is not zero and at location 1025_8 if there is not a skip when A is zero. An error can also occur at 1031_8 if a skip does not occur when A is zero, and if a skip occurs but A is not zero there will be an error indication at 1033_8 or 1035_8.

Skip if A Accumulator is Positive (SAPT)

B is incremented in the same manner as the zero test. An error at location 1045_8 means a skip should have occurred. An error at location 1056_8 indicates a skip occurred when A was not positive. A counter is used to test every case.

Skip A Accumulator is Negative (SANT)

Operates in the same manner as the A positive Test. An error at location 1071_8 indicates an illegal skip and an error at 1101_8 indicates no skip occurred.

Skip on A Accumulator Sign (SAST)

Runs similar to the previous tests except that there are three possibilities instead of two. An error at 1115_8 or 1117_8 indicates A was zero and the SAS did not detect this condition. An error at 1124_8 will occur when the SAS did not detect a positive sign. If a negative sign if not sensed, an error will occur at 1131_8.

Compare Memory to A Accumulator (CMAT)

The A Accumulator is loaded with the counter, a CMA to zero is executed and according to the skip after the CMA, the A Accumulator is tested for more, less, or equal to zero.

An illegal skip to n+1 will cause an error at location 1145_8.
An illegal skip to n+2 will cause an error at location 1150_8.
An illegal skip to n+3 will cause an error at location 1157_8.

Load and Store Instructions (LASA, LBSB)

The Accumulator is loaded with the counter and then stored in the location tagged STOP. A comparison between the stored data and the accumulator is then executed. The data is then compared with the counter. Errors at 1170_8 or 1172_8 indicate a bad STA, errors at 1174_8 or 1176_8 indicate a bad LAA. If an error occurs at location 1210_8 or 1212_8 the STB instruction failed, errors at 1215_8 or 1217_8 indicate LBA failed.

Transfer and Interchange A & B (TATB)

A is loaded with the counter and transferred to B, B is then stored and compared to A. An error at 1232_8 or 1234_8 indicates this phase failed.

B is loaded with the counter and is then transferred to A. A is compared to the counter and an error will occur at location 1241_8 or 1243_8 if TBA fails.

IAB is tested by loading A with the counter and B with minus one. After and IAB, A is compared to minus one, B is stored and A is loaded with STOR. A comparison then takes place. Errors at location 1250_8 or 1252_8 indicate a

bad LAA. If an error occurs at location 1210_8 the STB instruction failed, errors at 1215_8 or 1217_8 indicate LBA failed.

Transfer and Interchange A & B (TATB)

A is loaded with the counter and transferred to B, B is then stored and compared to A. An error at 1232_8 or 1234_8 indicates this phase failed.

B is loaded with the counter and is then transferred to A. A is compared to the counter and an error will occur at location 1241_8 or 1243_8 if TBA fails.

IAB is tested by loading A with the counter and B with minus one. After and IAB, A is compared to minus one, B is stored and A is loaded with STOR. A comparison then takes place. Errors at location 1250_8 or 1252_8 indicate A did not contain a minus one. If B did not contain the proper information errors will occur at 1256_8 or 1260_8.

Negate A Accumulator (NEGT)

The counter is subtracted from zero in A and stored. A is then loaded with the counter and negated. The results are compared and an error will occur at location 1274_8 or 1276_8 if they are not equal.

Shift Instructions (BEG1)

The RSA is tested extensively by loading A with a constant and shifting zero positions the first time. A comparison through an indirect address (DAT1, location 1343_8), checks the proper constant. The shift is incremented along with the indirect address. After all tests are completed, the shift and indirect address are returned to their original quantities. An error at 1306_8 or 1310_8 indicate an RSA error.

The rest of the shift instructions (SHTE) are tested two to six times, each shifting one position at a time. The results are compared to the proper constants.

Errors will occur at the following locations:

1352_8	LSA	1406_8	LSL
1354_8		1411_8	
1357_8		1414_8	
1361_8		1417_8	
1365_8			
1367_8		1423_8	RSA
1372_8		1435_8	
1374_8		1430_8	
1400_8		1432_8	
1402_8		1436_8	
		1440_8	
1451_8	RSL	1443_8	
1453_8		1445_8	
1456_8			
1460_8			

1465_8	FLA	
1467_8		
1472_8		
1474_8		
1500_8		
1502_8		
1505_8		
1507_8		
1514_8		
1516_8		
1521_8		
1523_8		
1531_8		
1533_8		
1540_8	FLL	
1542_8		
1545_8		
1547_8		
1554_8		
1561_8		
1566_8		
1573_8	FRA	
1575_8		
1600_8		
1602_8		
1606_8		
1610_8		
1613_8		
1615_8		
1622_8		
1624_8		
1627_8		
1631_8		
1635_8		
1637_8		
1642_8		
1644_8		
1651_8	FRL	
1653_8		
1656_8		
1660_8		
1664_8		
1666_8		
1671_8		
1673_8		

Add (ADD1, ADD3)

ADD1 - A is cleared, one is added to A and the counter is incremented, the results are compared and an error will occur at location 1737_8 or 1741_8 if there is a failure. This test is repeated in the B Accumulator. A failure in B is indicated by an error at location 1754_8 or 1756_8.

ADD3 - The next test adds the counter to itself in A and B, the registers are then loaded with the counter and shifted left one position. The sums are compared and errors will occur at location 1771_8 or 1773_8 for an error in A and 2010_8 or 2012_8 for an error in B.

Subtract (SUB1, SUB2, SUB3)

SUB1 - A is loaded with the counter, it is then subtracted, the A Accumulator is then checked for zero, a typeout at location 2023_8 indicates an error.

SUB2 - Zero is subtracted from the counter, A is then compared to the counter, if there is an error, a typeout will occur at location 2033_8 or 2035_8.

SUB3 - The counter is multiplied by two, it is then subtracted. A should then be equal to the counter, a halt at location 2046_8 or 2050_8 indicates A is not equal to the counter.

And A & B, Or A & B (ANOR)

Constants are anded and ored, the results are compared to constants. Error indications at locations 2057_8, 2061_8, 2075_8, 2077_8, 2113_8, 2115_8, and 2131_8 or 2133_8 are errors in the ABA instruction. Error indications at locations 2065_8, 2067_8, 2103_8, 2105_8, 2121_8, 2123_8, and 2137_8 or 2141_8 are errors in the OBA instruction. All worst cases are tested.

Increment B and Skip (IBST)

B is loaded with minus one and incremented, the counter is operated in the same fashion. The B Accumulator and the counter are compared. If no skip occurs while B is positive, an error will occur at location 2175_8, if B

skips when it is negative, an error will occur at location 2210_8. An unequal comparison between B and the counter will cause an error at location 2200_8 or 2202_8 when B is positive and at location 2213_8 or 2215_8 when B is negative.

Copy Sign of B - (CSBN, CSBP)

The CSB instruction is tested with the B sign bit on and off. With the bit on a CSB, NEG gets the bit into A, the counter is then negated and one is subtracted from it. The two answers are then compared. Error indications on this test are at locations 2234_8 and 2236_8.

With the B sign bit off a CSB, NEG is used again but the counter is only negated. After the comparison the error indications are at locations 2251_8 and 2253_8.

Complement Sign of A (ASCT)

The counter is loaded in A, the sign is changed by adding a minus sign. The counter's sign is then complemented. The two results are compared and if they are not equal, an error will occur at location 2267_8 or 2271_8.

Change Number Systems (CNST)

If the counter is negative, the data is checked, if the data is zero, nothing is done. All other cases the counter is changed by a CNS and a ASC, the counter is then negated. The results are compared and if they are not equal, an error will occur at location 2313_8 or 2315_8.

The remaining part of memory contains the cycle counter, error routine, and typeout routines.

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

 Catalog No. 303003A

7.5 Compare Memory to A, A Sign Test (CMASAS)

AUTHOR: SEL

ACCEPTED: 13 January 1967

PURPOSE: CMASAS tests every memory location with a CMA and SAS for every type of condition, except the first 410_8 locations. The reason for this test is that at certain program counter locations a CMA or SAS will not skip properly.

SOURCE PROGRAM LANGUAGE: MENMBLER 810A

COMPUTER CONFIGURATIONS: Standard SEL 810A

STORAGE: 0 to 411_8 plus every other memory location - Not relocatable

SUBROUTINES REQUIRED: 810A Mainframe Diagnostic Loading Procedure

TIMING: Dependent on memory size

USE: After loading set location 373_8 (TOP) with bits 2-3 dependent on memory size (see note). Start at location zero. CMASAS will run until manually halted.

NOTE: For a 4K memory -- set no bits in location 377_8.
8K - bit 3
12K - bit 2
16K - bits 2 & 3

Type-Out Formats:

L xxxxxx n
L = letter C for CMA error
letter S for SAS error
xxxxxx = the location of the erroneous instruction

n - a number, if a CMA error the number indicates the operand in memory, A is always zero. If an SAS error the number is what was contained in A. There are three possible numbers; 1, 0, -1.

NOTE: This program will clear every memory location.

METHOD: N/A

7.6 MEMDEX

AUTHOR: SEL

ACCEPTED: 13 January 1967

PURPOSE: Under sense switch control, the program will load into all memory locations; all zeros and ones, indirect and indirect indexed; alternate bits, indirect and indirect indexed; walking one, indirect; walking zero, indirect indexed. Each location is checked for the proper information stored.

SOURCE PROGRAM LANGUAGE: MNEMBLER 810A

COMPUTER CONFIGURATION: Standard SEL 810A

STORAGE: 0000 to 0502_8, plus every other memory location. Not relocatable.

SUBROUTINES REQUIRED: 810A Mainframe Diagnostic Loading Procedure

TIMING: Dependent on memory size.

USE: After loading, set the location tagged FIN (227_8) with the most significant four bits of the highest memory address (see note). Start at location zero. The program will run continuously until halted manually.

NOTE: For a 4K memory, no bits should be set in FIN.
8K - set bit 3
12K - set bit 2
16K - set bits 2 and 3

Snese Switches:

SSW 0 up - the dl ones, all zeros test will run.
SSW 1 up - the alternate bit pattern test will be run.

SSW 2 up - the walking one and walking zero test will run.
SSW 3 up - a halt will occur after an error type-out.

Any combination of sense switches may be used.

Type-Out Format:

12345 WORD aaaaaa
Memory Error
12345 - location at which the error occurred.

WORD - what the location should contain.
ZERO - if the location should contain a zero.
ONES - if the location should contain a zero.

1010 or 0101 - the sequence of binary bits for the alternate bit patterns.

1 or Z and XX - a walking one or zero error where XX = the left shift count from the farthest right position.

aaaaaa - the octal contents of the memory location in error.

NOTE: This program will destroy the contents of every memory location.

To restart this program, start at location 15_8.

METHOD:

Setup Routine

Sets the various addresses used to correspond with the highest memory address which is loaded into the location tagged FIN (227_8). FIN does not have to be changed if the machine in which the program is to be run has a 4096 location memory.

Sense Switch Routine (EXEC):

Checks the sense switches that are up and branches to the routine indicated by the sense switch settings.

All Ones, All Zeros Test (ALL1)

The zeros are obtained by clearing the A-Register. The

zeros are then stored and checked indirectly through the location tagged STAR (225_8). Ones are stored and checked indirectly through the location tagged FIN (227_8) which has its index bit set.

Alternate Bits Test (WORS)

The constant tagged ONEO (234_8) is stored and checked indirectly through STAR. The constant OH1 (235_8) is stored and checked indirectly through FIN.

Walking One and Zero Test (WALK)

The Walk One routine is executed first. A one is loaded in A and shifted zero times. The A-Register is then stored and checked indirectly through STAR. After all of memory is tested, the shift instruction is incremented and the test is repeated. When all bit positions are tested, the Walk Zero routine will be executed.

The Walk Zero routine is executed in the same manner as the Walk One routine except that FIN is used as an indirect address.

Ping Pong Routine (PIPO)

The starting addresses are changed to include the map not exercised previously. The routine then moves the entire program to either the top or bottom map and modifies itself to return the program to the map from which it was moved.

Address Reset Routine (REST)

This routine is executed after every test to reset STAR and the index count contained in the B-Register.

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

 Catalog No. 303005A

7.7	Load/Store/Register Change Test (LSRCT)
AUTHOR:	SEL
ACCEPTED:	13 January 1967
PURPOSE:	LSRCT uses each of the load, store and register change instructions except the LCS and CSB instruction. The data used is a counter, so all bit combinations are used. Errors are indicated by a type-out, as are successful cycles.
SOURCE PROGRAM LANGUAGE:	MNEMBLER 810A
COMPUTER CONFIGURATIONS:	Standard SEL 810A
STORAGE:	1000_8 to 1371_8 - Not Relocatable
SUBROUTINES REQUIRED:	810A Mainframe Diagnostic Loading Procedure
TIMING:	Approx. 2.5 ms per cycle
USE:	After loading start at location 1000_8. If an error occurs consult the error log to find what instruction failed.

Sense Switches:

SSW 0 up - the cycle count will not be typed
SSW 1 up - errors are ignored
SSW 2 up - no error type-out will occur, the machine will halt. The A Accumulator will contain the error location and locations 1256_8, 1257_8 and 1251_8 will contain the A&B Accumulators and the counter respectively.

Type-Out Formats:

Successful Cycles - NNNN
NNNN = the number of cycles completed without error.

Machine Error Preceeding Loc XXXXX
AAAAAA BBBBBB CCCCCC

XXXXX = the location plus one from which an SPB occurred following an error condition.
AAAAAA = the contents of the A Accumulator
BBBBBB = the contents of the B Accumulator
CCCCCC = the contents of the counter

The locations listed are what will be typed-out if an error occurs.

Location	Op Code
1003	CLA
1007, 1011	LBA, TBA
1014, 1016	IAB
1021, 1023	LAA
1027, 1031	TAB, STB
1034, 1036	STA

METHOD: N/A

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

 Catalog No. 303006A

7.8	Arithmetic Test (ADDO)
AUTHOR:	SEL
ACCEPTED:	13 January 1967
PURPOSE:	This program exercises the adder using the AMA, AMB and SMA instructions. RNA is also tested. A random bit pattern generator is used to generate operands. Memory is added to A and B, the results are compared. Memory is subtracted from A using the same operands, one in A, one in memory, then visa versa. The differences are compared ignoring the signs. RNA is tested by a software round A simulation. Over flow is checked and an error condition will be generated if the overflow latch is not set at the proper time.
SOURCE PROGRAM LANGUAGE:	MNEMBLER 810A
COMPUTER CONFIGURATIONS:	Standard SEL 810A
STORAGE:	1000_8 to 1441_8 - Not relocatable
SUBROUTINES REQUIRED:	810A Mainframe Diagnostic Loading Procedure
TIMING:	Approx. 400 microseconds/cycle if no errors occur.
USE:	Start at location 1000_8. The program will run until manually halted. Sense Switches SSW 0 up - Errors are ignored SSW 1 up - A halt will occur after an error type-out. SSW 2 up - No error type-out, a halt will occur NOTE: With SSW 2 up a halt at location 1131 indicates an RNA error. An add error will cause a halt at 1213_8 and a subtract error halts at 1264_8.

Type-Out Formats:

```
aaaaaa   bbbbbb
A  nnnnnn OVFL
B  mmmmmm  OVFL
```

Indicates add error:

aaaaaa = operand in A for AMA, in memory for AMB
bbbbbb = operand in memory for AMA, in B for AMB
nnnnnn = the AMA sum
mmmmmm = the AMB sum

NOTE: If both sums are the same and the letters OVFL (indicating overflow) are not typed next to both sums this indicates an overflow error. The letters will not always be typed, only if an overflow occurred.

```
aaaaaa   bbbbbb
S A   nnnnnn  OVFL
   B   mmmmmm  OVFL
```

Indicates an SMA error:

nnnnnn = difference of a-b
mmmmmm = difference of b-a

NOTE: Only the signs should be unlike. As in the add test overflow should occur on both subtracts.

```
aaaaaa   bbbbbb
 R nnnnnn   mmmmmm
```

Indicates an RNA error:

a's = A Accumulator
b's = B Accumulator
nnnnnn = software RNA
mmmmmm = hardware RNA

METHOD: N/A

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

 Catalog No. 303007A

7.9 Multiply Test (MTPY)

AUTHOR: SEL

ACCEPTED: 13 January 1967

PURPOSE: MTPY uses a random operand generator to generate two operands. The two operands are multiplies by the hardware, the product is then compared to the product of a software multiply. An inequality causes a typeout. The software multiply arrives at a product by adding and shifting.

SOURCE PROGRAM LANGUAGE: MNEMBLER 810A

COMPUTER CONFIGURATIONS: Standard SEL 810A

STORAGE: 1000_8 to 1270_8 - Not Relocatable

SUBROUTINES REQUIRED: 810A Mainframe Diagnostic Loading Procedure

TIMING: Approx. 0.75 ms/product if no error occurs

USE: The program will run until halted manually.

Sense Switches

SSW 0 up - errors are ignored
SSW 1 up - no error type-out, a halt will occur
SSW 2 up - the same operands will be used continuously
SSW 3 up - a halt will occur after an error type-out

NOTE: If it is desired to find two operands that fail continuously set sense switch 3 up, after the type-out and halt set sense switches 0 up and 2 up and 3 down. The program will run continuously using the operands that failed and the error condition will be ignored allowing easier trouble shooting.

Type-Out Format:

Multiply Error

```
aaaaaa      bbbbbb
 nnnnnn      mmmmmm
 xxxxxx      yyyyyy
```

aaaaaa = multiplier (in memory)
bbbbbb = multiplicand (in B Accumulator)
nnnnnn = Software product in A
mmmmmm = Software product in B
xxxxxx = Product in A
yyyyyy = Product in B

METHOD: N/A

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

 Catalog No. 303008A

7.10	Divide
AUTHOR:	SEL
ACCEPTED:	13 January 1967
PURPOSE:	Divide uses a software divide which simulates the hardware divide exactly. Both hardware and software divide operands in single and double precision forms, the quotients and remainders are compared for accuracy.
SOURCE PROGRAM LANGUAGE:	MNEMBLER 810A
COMPUTER CONFIGURATIONS:	Standard SEL 810A with divide instruction -
STORAGE:	1000_8 to 1524_8 - Not relocatable
SUBROUTINES REQUIRED:	810A Mainframe Diagnostic Loading Procedure
TIMING:	Approx. 1050 microseconds/cycle without errors
USE:	Start at location 1000_8. The program will run until manually halted.

Sense Switches

SSW 0 up - Errors are ignored
SSW 1 up - No error type-out will occur, the machine will halt
SSW 2 up - The same operands will be used continuously
SSW 3 up - A machine halt will occur after the error type-out.

NOTE: To find operands that fail set 3 up, after the halt set 0 and 2 up, this will repeat the operands and errors will be ignored which will aid trouble shooting.

Type-Out Format:

```
xxxxxx   yyyyyy
 aaaaaa  bbbbbb
 cccccc  dddddd
```

Single precision divide error:

xxxxxx = B Accumulator operand
yyyyyy = memory operand
aaaaaa = quotient, software
bbbbbb = remainder, software
cccccc = quotient, hardware
dddddd = remainder, hardware

```
mmmmmmm  nnnnnn  xxxxxx
 aaaaaa  bbbbbb
 cccccc  dddddd
```

Double precision divide error:

mmmmmmm = A Accumulator operand
nnnnnn = B Accumulator operand
xxxxxx = memory operand
a's, b's, c's, d's = same as single precision

NOTE: If the letters "OVFL" are typed out on a double precision divide error in place of a quotient and remainder, this indicates that operation caused a divide overflow. The hardware should get overflow when the software does and the hardware should not get overflow when the software does not.

METHOD: N/A

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

7.11	Memory Worst Case Test (MEMTES)
AUTHOR:	SEL
ACCEPTED:	13 January 1967
PURPOSE:	MEMTES analyzes the program counter bits in conjunction with a Boolean expression to find which locations should be loaded with ones or zeros. After all memory is loaded, each location is unloaded sequentially. While unloading memory, the worst case pattern will cause additive noise in the sense windings possibly causing bits to be dropped or picked up. All of memory is tested through the use of a ping-pong routine. After the upper portion of memory has been exercised (location 1000_8 and up), the program is modified to exercise the lower portion of memory (location 0 up to, but not including the highest map) and transferred to the highest map in memory. Once the lower protion is exercised, the program is reset to exercise upper memory and moved back to the lowest map.
SOURCE PROGRAM LANGUAGE:	MNEMBLER 810A
COMPUTER CONFIGURATIONS:	Standard SEL 810A
STORAGE:	0000 to 0467_8, plus every other location - Not relocatable
SUBROUTINES REQUIRED:	810A Mainframe Diagnostic Loading Procedure
TIMING:	Dependent on memory size.

USE:

After loading, set the location tagged FIN (420_8) with the four (4) most significant bits of the highest memory address (see note).

Set the sense switches to the desired combination before starting.

Start at location zero. The program will run continuously until halted manually.

NOTE: For a 4K memory, no bits should be set in FIN
8K - set bit 3
12K - set bit 2
16K - set bit 2 and 3

Sense Switches:

Set - No switches for: Ferroxcube, 4K Memory
SSW 0 up for: Ferroxcube, 8K Memory
SSW 0 and 1 up for: Ampex Mod 1, 8K Memory
SSW 1 up for: Ampex Mod 1, 4K Memory
SSW 2 up for: Ampex Mod 2, All Memories

NOTE: Be sure the proper sense switches are set before the program is started, otherwise the wrong worst case will be used.

Type-Out Format:

aaaaa b ccccccccccccccc

Memory Unload Error

a's = octal memory location in error.
b = a one or a zero, what every bit position of the error location should contain.
c's = sixteen binary bits which were unloaded from the error location.

NOTE: A parity error may also be caused when unloading a location. If a parity error occurs, there may not be an error type-out. The A-Register may be displayed and if it does not contain either all ones or all zeros, the parity error may be cleared and the program started where it has stopped, the error type-out will follow. If, however, the A register does contain all ones or all zeros, the B Register may be displayed to find the location that caused the parity error.

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

7.12	810A Paper Tape Reader/Punch Test
AUTHOR:	J B. Boyer, SEL
ACCEPTED:	17 January 1967
PURPOSE:	This test is designed to verify the correct operation of a high speed paper tape system.
SOURCE PROGRAM LANGUAGE:	MNEMBLER
COMPUTER CONFIGURATION:	SEL 810A with high speed paper tape punch and reader
STORAGE:	Less than one map - Full above location 2000_8
SUBROUTINES REQUIRED:	N/A
TIMING:	31 seconds per test cycle (excluding error typeout)
USE:	Load the program using the Standard Relocatable Loader.

Operation:

Sense Switch Settings:

Switch 0 up - terminate test at end of present cycle
down - start another cycle after present cycle

Switch 1 up - eliminate type out of errors
down - type message when error detected

Switch 2 up - restart test after detection of an error
down - continue test after error

Control switch 15 is set to eliminate all typeout.

1. Start at location 2000_8. The program will type
"PAPER TAPE READER/PUNCH TEST"
"DATE"

The operator must then type the date and when finished, type a carriage return character. The program will then turn on the punch and punch two forward and reverse binary progressions, turn off the punch and type "PUT TAPE INTO READER THEN PRESS START".

2. When the operator complies with these instructions, the program will commence punching and reading until either sense switch 0 is set or the program detects an error.

3. When an error is detected, the program will type
"ERROR CONDITION DETECTED SHOULD HAVE READ XXX
DID READ XXX"

The program will then halt. The operator has the option of having the test continue or restart (sense switch 2). If switch 1 had been set the error would be counted but otherwise ignored.

4. Upon termination (sense switch 0) the program will type "NUMBER OF CYCLES COMPLETED XXX
"NUMBER OF ERRORS DETECTED XXX"
it will then turn off punch power and halt.

METHOD:

This test is designed to exercise the mechanical functions of the punch and reader as well as test the accuracy of data transfer. To test the mechanical functions of punch and reader they are run in three modes:

1. continuous mode,
2. start/stop mode
3. simultaneous mode

In the continuous mode the punch and reader are operated at their maximum rates. In start/stop mode there is a delay of .02 seconds between each character punched and read. In the simultaneous mode the punch and reader are operated simultaneously.

To test the data transfer the program punches and reads repeated forward and reverse binary progressions, going from 001 to 377_8, then 000, then 377_8 back to 001. A test

cycle is one forward and reverse progression punched and read in each of the three modes. The program will keep repeating the test cycle until the operator terminates it or an error is detected. When an error is detected, a message is typed on the ASR-33 indicating what should have been read and what was read. The operator can determine by looking at the tape whether the reader or punch was in error. The operator then has the option to either restart the test or continue. The operator also has the option to eliminate type out of errors, but they will be counted and the number of errors will be typed out upon termination of the test.

SEL PROGRAM LIBRARY

PROGRAM DESCRIPTION

 Catalog No. 303012A

7.13 ASR 33/35 Teletype Test

AUTHOR: SEL

ACCEPTED: 10 February 1967

PURPOSE: To provide a diagnostic program for the ASR 33 and ASR 35 Teletypes

COMPUTER CONFIGURATION: Basic 810A

SUBROUTINES REQUIRED: None

STORAGE: Relocatable with a bias of 2000_8. 455_8 memory locations

TIMING: N/A

USE: Load the relocatable object tape by means of the 810A Binary Relocatable Loader (Cat. No. 300002A).

Sense Switch 0 up.

Test 1a
A binary progression will be punched. A halt occurs at location 2057 to load punched tape into reader.

Test 1b
Depress start and tape will be read, compared, and duplicated. A halt will occur at location 2146 to load duplicated tape into reader.

Test 1c
Depress start for high speed read and compare.

NOTE: Any time program halts at location 2361 an error has occurred. Depress start to continue.

The keyboard turn-around test is executed. Input from keyboard is outputted on page printer.

Sense Switch 2 up

Reader will read tape on interrupt basis and punch will duplicate on interrupt basis.

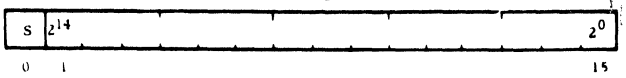
Sense Switch 3 up

All errors will be ignored and a continuous load will be executed during test 1b and 1c.

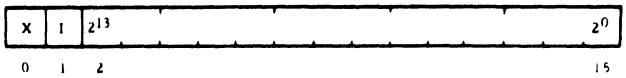
APPENDIX A

COMPUTER WORD FORMATS

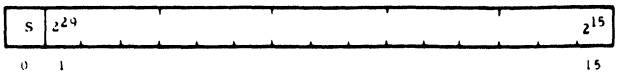
Integer Data

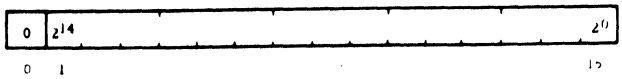


Indirect Address Word

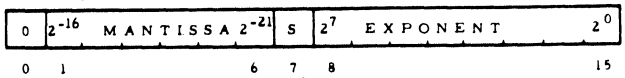


Double-Precision Fixed Point Data

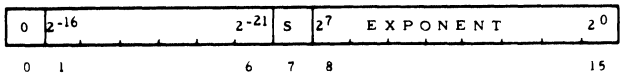


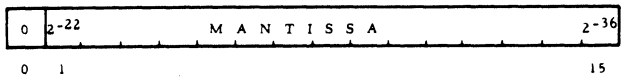


Single-Precision Floating Point Data

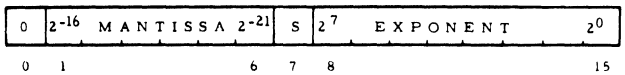


Double-Precision Floating Point Data

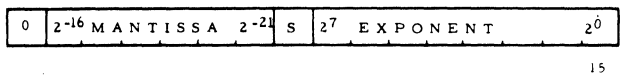




Complex Floating Point Data

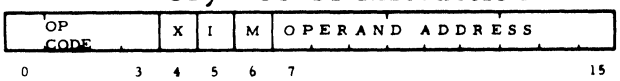


(Real Part)

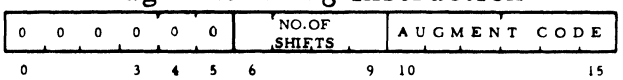


(Imaginary Part)

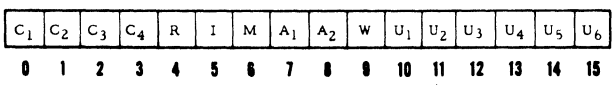
Memory Access Instruction



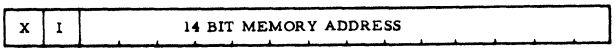
Augmented 00$_8$ Instruction



Input/Output Instructions

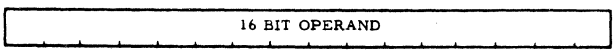


First Word



Second Word (Address Mode)

OR



OR

Second Word (Immediate Mode)

SEL 810A Computer Word Formats

APPENDIX B

ALPHA CHARACTERS

Character	Teletype ASR-33	Teletype ASR-35	Line Printer (Truncated ASC II)	IBM/BCD	Hollerith Card Code
A	301	101	01	61	12-1
B	302	102	02	62	12-2
C	303	303	03	63	12-3
D	304	104	04	64	12-4
E	305	305	05	65	12-5
F	306	306	06	66	12-6
G	307	107	07	67	12-7
H	310	110	10	70	12-8
I	311	311	11	71	12-9
J	312	312	12	41	11-1
K	313	113	13	42	11-2
L	314	314	14	43	11-3
M	315	115	15	44	11-4
N	316	116	16	45	11-5
O	317	317	17	46	11-6
P	320	120	20	47	11-7
Q	321	321	21	50	11-8
R	322	322	22	51	11-9
S	323	123	23	22	0-2
T	324	324	24	23	0-3
U	325	125	25	24	0-4
V	326	126	26	25	0-5
W	327	327	27	26	0-6
X	330	330	30	27	0-7
Y	331	131	31	30	0-8
Z	332	132	32	31	0-9
		Numerics			
0	260	060	60	12	0
1	261	261	61	01	1
2	262	262	62	02	2
3	263	063	63	03	3
4	264	264	64	04	4
5	265	065	65	05	5
6	266	066	66	06	6
7	267	267	67	07	7
8	270	270	70	10	8
9	271	071	71	11	9

(Special Characters on Sheet 2)

Special Symbols

Character	Teletype ASR-33	Teletype ASR-35	Line Printer (Truncated ASC II)	IBM/BCD	Card Code
@	300	300	00	17	8-7
⌈	333	333	33	55	11-8-5
\	334	134	34	57	11-8-7
⌉	335	335	35	36	0-8-6
↑	336	336	36	32	0-8-2
←	337	137	37	77	12-8-7
Space	240	240	40	20	
!	241	041	41	52	11-0
"	242	042	42	37	0-8-7
#	243	243	43	35	0-8-5
$	244	044	44	53	11-8-3
%	245	245	45	75	12-8-5
&	246	246	46	72	
'	247	047	47	14	8-4
(	250	040	50	34	0-8-4
)	251	251	51	74	12-8-4
*	252	252	52	54	11-8-4
+	253	053	53	60	12
,	254	254	54	33	0-8-3
-	255	055	55	40	11
.	256	056	56	73	12-8-3
/	257	257	57	21	0-1
:	272	072	72	15	8-5
;	273	273	73	56	11-8-6
	274	074	74	76	12-8-6
=	275	275	75	13	8-3
	276	276	76	16	8-6
?	277	077	77	00	12-11
				57	
≠				32	
Carriage Return	215	215			
Line Feed	212	012			
Bell	207	207			
Delete	377	377			

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
MAGNETIC TAPE FORMAT 0		CONNECT = 1 DISCONN = 0	WORD TRANSFER READY INTERRUPT	END OF RECORD INTERRUPT	(FORMAT) 0	REWIND	ERASE 4 INCHES OF TAPE	BCD = 1 BINARY = 0	DENSITY		TAPE TRANSPORT			CURRENT WORD ADDRESS IN	CHARACTERS PER WORD	
MAGNETIC TAPE FORMAT 1	BTC INITIALIZE		WORD TRANSFER READY INTERRUPT	END OF RECORD INTERRUPT	(FORMAT) 1	WRITE RECORD	WRITE END OF FILE	READ RECORD	ADVANCE RECORD	ADVANCE/ END OF FILE	BACKSPACE RECORD	BACKSPACE END OF FILE	CORRECT CRC ERROR (9 TRACK OPTION)	CURRENT WORD ADDRESS IN	CAB	
ASR-33/35			IN	OUT	READER MODE	KEY MODE	CLEAR									
PAPER TAPE READER AND PUNCH			IN	OUT	PUNCH POWER ON	PUNCH POWER OFF	READER ENABLE	READER DISABLE								
NCR CARD READER			IN		BINARY FEED CARD	BCD FEED 1 CARD		REAL STACK OFFSET	EJECT CARD (PUNCH)	PUNCH STACKER OFFSET						
CALCOMP			END OF EX.		PEN DOWN	PEN UP	DRUM DOWN	DRUM UP	CAR. LEFT	CAR. RIGHT						
LINE PRINTER			END OF PRINT	BUFF NOT BUSY	ADVANCE FORMAT N	ADVANCE 1 LINE	TOP OF DRUM	PRINT	CLEAR BUFFER	FILL BUFFER						
DISC SEEK			SEEK ERROR	SEEK COMPLETE		NUMBER OF TRACKS TO BE MOVED							1		*FWD	*REV
						64	32	16	8	4	2	1				
DISC DATA			SEEK ERROR	SEEK COMPLETE		SECTOR NUMBER				HEAD NUMBER			0		WRITE	READ
					8	4	2	1	8	4	2	1				

CEU Second Word Format

*TO SEEK TRACK 00 BOTH BITS MUST BE ZERO

CARD READER AND PUNCH										SKIP NO PUNCH ERROR						
DISC					SKIP SEEK COMP	SKIP NO SEEK ERROR	SKIP ON BOD	SKIP ON BOS	SKIP PACK ON LINE	SKIP NO READ OVERFLOW	SKIP NO WRITE OVERFLOW	SKIP NO CHECK SUM ERROR	SKIP NO FILE UNSAFE	SKIP DCU READY	SKIP NOT BUSY	
MAGNETIC TAPE	SKIP ON NOT BUSY	SKIP ON NO END OF FILE	SKIP ON NO OVERFLOW	SKIP ON LOAD POINT	SKIP ON END OF RECORD INTERRUPT	SKIP ON NO PARITY ERROR	SKIP ON WRITE RING IN	SKIP ON NO END OF TAPE	SKIP ON REWINDING	SKIP ON NO CRC ERROR (9 TRACK ONLY)						
LINE PRINTER					SKIP NOT BUSY		SKIP NO PARITY ERROR		SKIP NO BOTTOM OF FORM	SKIP IF PRINTER OP.						

TEU Second Word Format

A2082

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
MAGNETIC TAPE FORMAT 0	BTC INITI-ALIZE	PIE = 1 PID = 0	WORD TRANS-FER INTER-RUPT	END OF RECORD INTER-RUPT	FORMAT 0	REWIND	ERASE FOUR INCHES OF TAPE	BCD = 1 BINARY = 0									DENSITY		TAPE TRANSPORT			CURRENT WORD ADDRESS IN	CHARACTERS PER WORD	
MAGNETIC TAPE FORMAT 1			WORD TRANS-FER INTER-RUPT	END OF RECORD INTER-RUPT	FORMAT 1	WRITE RECORD	WRITE END OF FILE	READ RECORD									ADVANCE RECORD	ADVANCE END OF FILE	BACK-SPACE RECORD	BACK-SPACE END OF FILE	CORRECT CRC ERROR (9 TRACK OPTION)			
ASR-33			IN	OUT	READER MODE	KEY MODE	CLEAR																	
PAPER TAPE READ AND PUNCH			IN	OUT	PUNCH POWER ON	PUNCH POWER OFF	READER ENABLE	READER DISABLE																
CARD READER			IN		FEED CARD																			
CALCOMP			END OF EX.		PIN DOWN	PIN UP	DRUM DOWN	DRUM UP									CAR. LEFT	CAR. RIGHT						
LINE PRINTER			END OF PRINT	BUFF NOT BUSY	ADVANCE FORMAT N SEE NOTE	ADVANCE 1 LINE	TOP OF FORM	PRINT									CLEAR BUFFER	FILL BUFFER						
SELECTRIC																								
AR/AN-35																								
DISC																								
			INT1	INT2																				

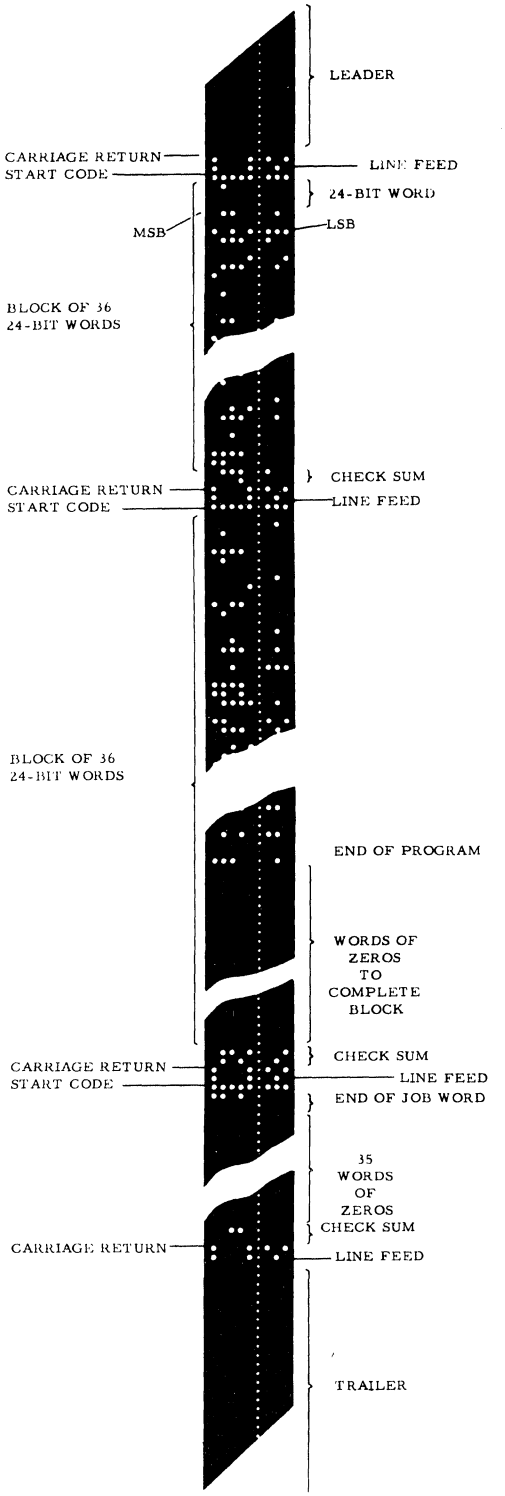
NOTE: WHEN A ONE IS PRESENT IN BIT POSITION 4, ADVANCE TO THE CHANNEL NUMBER (EXPRESSED IN OCTAL) REPRESENTED BY THE BITS PRESENT IN POSITIONS 7, 16 AND 17.

BITS	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
MAGNETIC TAPE	SKIP ON NOT BUSY	SKIP ON NO END OF FILE	SKIP ON NO OVER-FLOW	SKIP ON LOAD POINT	SKIP ON END OF RECORD INTER-RUPT	SKIP ON NO PARITY ERROR	SKIP ON WRITE RING IN	SKIP ON NO END OF TAPE									SKIP ON REWIND-ING	SKIP ON NO CRC ERROR (9 TRACK ONLY)						
LINE PRINTER					SKIP NOT BUSY		SKIP NO PARITY ERROR										SKIP NO BOTTOM OF FORM	SKIP IF PRINTER OP.						

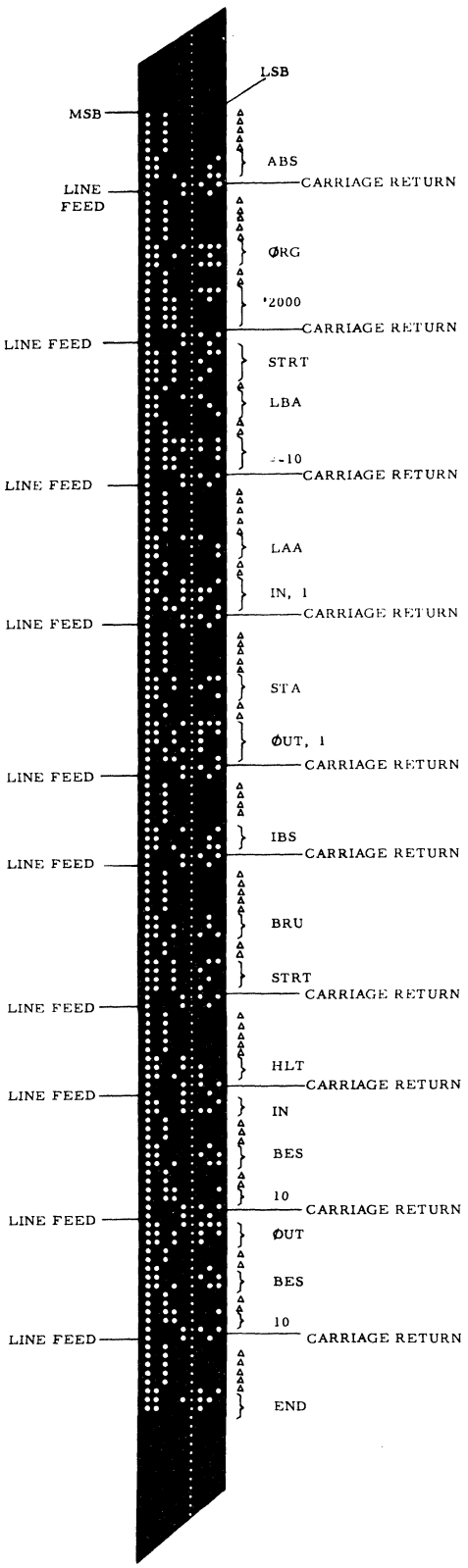
A1852

APPENDIX C
CEU AND TEU SECOND WORD FORMATS

APPENDIX D



SEL 810A ASSEMBLER AND COMPILIER OBJECT PROGRAM OUTPUT TAPE. MUST BE LOADED WITH THE SEL MNEMBLER LOADER PROGRAM.



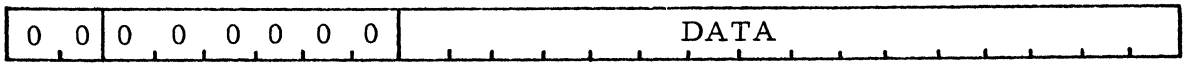
ASC II CODE ASSEMBLER SOURCE INPUT IN CARD FORMAT IMAGE. MUST BE LOADED BY ASSEMBLER.

A2198

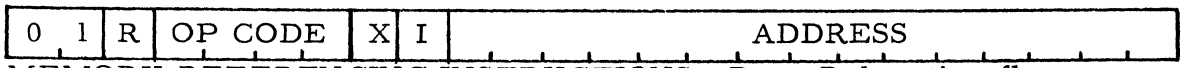
SEL 810A Paper Tape Formats

APPENDIX E

SEL 810A Assembler Output Formats

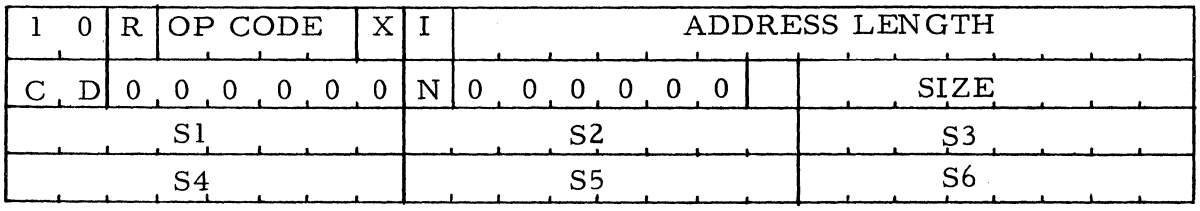


DIRECT LOAD: Data or Non-memory-referencing instructions.



MEMORY REFERENCING INSTRUCTIONS: R = Relocation flag
(DAC) OP = '13, 14-bit address constant
(EAC) OP = '17, 15-bit address constant

LITERAL REFERENCING INSTRUCTIONS:

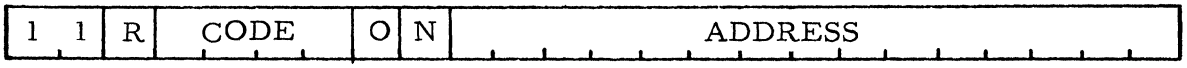


SUBROUTINE OR COMMON:
CD = 10: Common defn.
Address = length

= 11: Common request
Address = rel. to block.
N = negative flag

CD = 00: Subroutine definition (NAME)
Address = relative entry point.

= 01: Subroutine call (CALL)
Address = 0



SPECIAL ACTION:
Code = 00, Establish Load Point
= 01, END Jump
= 02, STRING
= 03, 9-Bit ADD-TO
= 04, 14-Bit ADD-TO (DAC)
= 05, 15-Bit ADD-TO (EAC)
= 06, Turn on CHAIN flag
= 07, Turn on Load flag
= 10, END-OF-JOB

APPENDIX F

CLASS	MNEMONIC	OP CODE	CYCLES (1.75 Microseconds)	FUNCTION
ARITHMETIC:	AMA	05	2	Add Memory to A
	AMB	16	2	Add Memory to B
	SMA	06	2	Subtract Memory from A
	MPY	07	5	Multiply B times Memory
	DIV*	10	7	Divide A and B by Memory
	RNA'	00-01	1	Round A by MSB in B
LOAD/STORE:	LAA	01	2	Load A from Memory
	LBA	02	2	Load B from Memory
	STA	03	2	Store Memory from A
	STB	04	2	Store Memory from B
	LCS'	00-31	1	Load Control Switches in A
BRANCH/SKIP:	BRU	11	1	Unconditional Branch
	SPB	12	2	Store Place and Branch
	SNS	13-4	1	Skip if Control Switch Not Set
	IMS	14	3	Increment Memory and Skip if 0
	CMA	15	2	Compare Memory and A (3 Way) n+1 if (A)-(M) < 0, n+2 if (A)=(M) n+3 if (A)-(M) > 0
	IBS'	00-26	1	Increment B (Index) and Skip if 0
	SAZ'	00-22	1	Skip if A is Zero
	SAP'	00-24	1	Skip if A is Positive
	SAN'	00-23	1	Skip if A is Negative
	SOF'	00-25	1	Skip No Overflow
	SAS'	00-21	1	Skip on A Sign (3 Way) n+1(-), n+2(0), n+3(+)
	SNO'	00-32	1	Skip if A is Normalized
	LOB'	00-36	2	Long branch
LOGICAL:	ABA'	00-27	1	AND A and B
	OBA'	00-30	1	OR A and B
	NEG'	00-02	1	Negate A
	ASC'	00-20	1	Complement A Sign
	CNS'	00-34	1	Convert Number System
REGISTER CHANGE:	CLA'	00-03	1	Clear A
	TAB'	00-05	1	Transfer A to B
	IAB'	00-06	1	Interchange A and B
	CSB'	00-07	1	Transfer B sign to Carry and Clear B Sign to Positive
	TBA'	00-04	1	Transfer B to A
SHIFT:	RSA'	00-10	Time for Shifts vary as follows:	Right Shift A
	LSA'	00-11		Left Shift A
	FRA'	00-12	Shifts / Time	Right Shift A and B
	FLA'	00-17		Left Shift A and B
	RSL'	00-15	1-4 / 2	Right Logical Shift A
	FRL'	00-14	5-8 / 3	Full Rotate Logical A and B Left
	LSL'	00-16	9-12 / 4	Left Logical Shift A
	FLL'	00-13	13-15 / 5	Left Logical Shift A and B
CONTROL:	HLT'	00-00	1	Halt
	NOP'	00-33	1	No Operation
	TOI'	00-35	1	Turn off Interrupt
	PIE'	130600	2	Enable Interrupt
	PID'	130601	2	Disable Interrupt
INPUT/OUTPUT:	CEU'	13.0IM.0	3	Command External Unit
	TEU'	13.0IM.2	3	Test External Unit
	AOP'	1700	3	Accumulator Word Out to Unit
	AIP'	1702	3	Accumulator Word In from Unit
	MOP'	17.0IM.4	3	Memory Word Out to Unit
	MIP'	17.0IM.6	3	Memory Word In from Unit
OPTIONS:	PON' *	002040	2	Protect Bit On
	POF' *	002041	2	Protect Bit Off
	TBV' *	00-42	1	Transfer B Register to VBR
	TVB'*	00-43	1	Transfer VBR to B Register

* Optional
' Augmented