CONTROL DATA

160-A COMPUTER

160-A

OSAS-A/160-A ASSEMBLY SYSTEM

OSAS-A/160-A ASSEMBLY SYSTEM

This manual, publication 60050700, Revision C, is a major revision and obsoletes publication 507B. Any comments should be addressed to:

Control Data Corporation
Documentation and Evaluation Department
3145 Porter Drive
Palo Alto, California

April, 1964
Pub. No. 60050700
Revision C

©Control Data Corporation
Printed in the United States of America

CONTENTS

CONTENTS (CONT'D)

INTRODUCTION

The 160-A Assembly System (OSAS-A) allows a programmer to write programs for the CONTROL DATA* 160-A Computer in symbolic notation with minimum regard for ultimate storage locations assigned a program. The assembly system also allows a programmer to write instructions using mnemonic symbols, which are easier to remember and to later interpret than are the octal machine code equivalents.

OSAS-A assembles symbolic programs from cards, paper tape or magnetic tape. For each program assembled, OSAS-A provides both a listable and a binary output on any of the three media - cards, paper tape or magnetic tape. In addition, listable output may appear on a line printer. The assembly system can be utilized by any 160-A configuration, ranging from a system that consists of the 160-A computer and paper tape input/output equipment to the full range of paper tape, magnetic tape, card and printer equipment that may be included in a 160-A computer system.

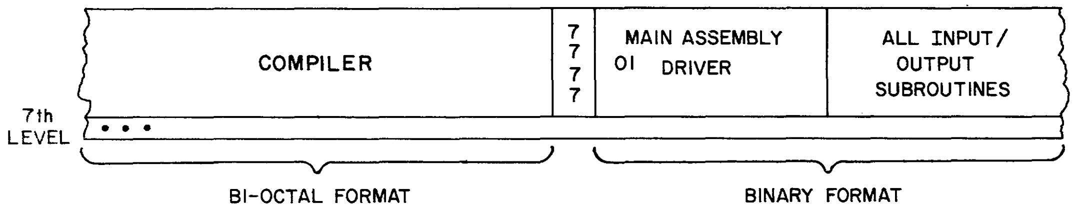
The OSAS-A manual is intended to be used in conjunction with the 160-A programming manual.

*Registered trademark of Control Data Corporation

PART I

MASTER PAPER TAPE

The complete 160-A assembly system is contained on a Master Paper Tape (MPT). This tape contains the OSAS-A compiler, the main assembly driver and all input/output subroutines. The function of the compiler is to produce a working version of the MPT that consists of the main assembly driver and the input/output subroutines designated by the programmer. The main assembly driver is the assembly program. The I/O subroutines control the type of symbolic input, intermediate input/output, and listable and binary output. The working version of the MPT is compiled and stored in core, and may also be punched on paper tape for future assemblies using the same I/O configuration.



The compiler is in bi-octal format and comprises the first record on the MPT; the remainder of the routines on the MPT are in binary format and form the second record. To assemble a symbolic program, the compiler is machine loaded; as soon as it is stored in core, an input parameter specifying the required I/O subroutines is manually entered in the A register and the compiler is run. At the completion of the compiler run, the required working version of the MPT is in core and on paper tape, if specified.

The MPT can be updated by adding or deleting routines in the second record; the master paper tape edit program can also duplicate the MPT as described in Part III, Master Paper Tape Edit.

INPUT PARAMETER

I/O FORM	PARAMETER	MEDIA	EQUIPMENT
Symbolic Input	0XXX	Paper Tape	
	1XXX	Magnetic Tape	163
	2XXX	Cards	Fast Reader via 1610
	3XXX	Cards	167-1
	4XXX	Cards	167-2
	5XXX	Magnetic Tape	1608 (Low Density)
	6XXX	Magnetic Tape	1607
	7XXX	Cards	405
Intermediate Input/Output	X0XX	Paper Tape	
	X1XX	Magnetic Tape	163 Assembly Mode
	X4XX	Magnetic Tape	163 Character Mode
	X5XX	Magnetic Tape	1608 (Low Density)
	X6XX	Magnetic Tape	1607
Listable Output	XX0X	Paper Tape	
	XX1X	Magnetic Tape	163
	XX2X	Line Printer	166
	XX3X	Line Printer	1612
	XX5X	Magnetic Tape	1608 (Low Density)
	XX6X	Magnetic Tape	1607
Binary Output	XXX0	Paper Tape	
	XXX1	Magnetic Tape	163 Character Mode
	XXX2	Cards	Faster Reader via 1610
	XXX5	Magnetic Tape	1608 (Low Density)
	XXX6	Magnetic Tape	1607

Entering an improper parameter will cause the MPT to stop at 7140 with the incorrect parameter in the A register. To continue, reload the compiler, enter the correct parameter in the A register, and run from P = 7000.

RECORD ONE

This portion of MPT consists of a Compiler and forms the first physical record on the tape. It is in bi-octal format with a 7777 code at the end of the record. MPT will stop after this record is read and wait for the input parameter. MPT operation instructions are given at the end of Part I.

Compiler

The Compiler loads the proper I/O subroutines for the version of OSAS-A designated by the input parameter. It also generates a bi-octal tape if requested.

The ID numbers are used by the compiler to identify I/O subroutines during compilation of the working version of the MPT. These numbers are also used by the MPT edit program for locating routines in the second record.

RECORD TWO

This portion of the MPT contains the Main Assembly Driver and all current I/O subroutines. It is in binary format and forms the second record on the MPT. The Main Assembly Driver and the I/O subroutines loaded by the compiler constitute the working version of OSAS-A. The I/O configuration of any working version is specified by the input parameter. As a version of OSAS-A is generated, it may be punched on a bi-octal paper tape. Loading the paper tape is faster than regenerating the assembly program each time by means of the MPT and input parameter.

Main Assembly Driver

This program is preceded by a 01 on the MPT. The Main Assembly Drive, which consists of the main assembly logic, controls execution of designated I/O subroutines during assembly of a symbolic program.

Input/Output Subroutines

Each subroutine is identified by a two digit prefix and controls a specific type of input or output.

BI-OCTAL TAPE OF OSAS-A ASSEMBLY SYSTEM

Bi-octal tapes of OSAS-A produced by the compiler are not in relocatable form. Leader for each tape is generated automatically by the compiler when the tape is punched.

OPERATING INSTRUCTIONS FOR MPT

To generate a version of OSAS-A without bi-octal tape

1. Machine load MPT at location 7000, relative bank zero; reader stops with 7647 in P register and 0435 in A register. Compiler now loaded.
2. Leave tape positioned in reader; reader on.
3. Master clear; enter input parameter into A register, 7000 into P register and run.
4. When the program stops at 7151, OSAS-A is in bank zero. Clear and run to assemble programs using this I/O configuration.

To generate a version of OSAS-A with bi-octal tape

1. Generate internal version of OSAS-A as described above.
2. When program stops at 7151, turn punch on and run. DO NOT CLEAR.
3. When the program stops at 7172, OSAS-A has been punched correctly. OSAS-A is in bank zero and also on bi-octal paper tape; programs using this version can now be assembled. Similar programs can be assembled later by simply loading the taped version.
4. Clear and run to assemble.

To load taped version of OSAS-A

1. Clear core, master clear, set all bank storage registers to zero, and load bi-octal tape at location 0000.
2. Master clear and run.

Stops

7140	Invalid input parameter; enter correct parameter into A register and run.
7151	Generation of OSAS-A is complete. 1. To assemble: clear and run. 2. To punch a bi-octal tape of OSAS-A: run.
7172	OSAS-A punched correctly. 1. To assemble: clear and run.

7372	Bad master tape caused by longitudinal parity error. (The word count in the binary card image does not agree with the number of characters encountered.) Correct MPT and reload compiler.*
7447	Error on master tape caused by 7, 9 punches missing from column 1 in binary card image. One of these punches may be missing or misread, in which case ignore error and continue by running. If a character is bad, master tape is bad and other error stops will occur.
7470	Check sum error on a transfer card image. Recommend reloading MPT. Ignore and continue by running if desired.
7600	Longitudinal check sum error. The check sum as part of the binary card image does not agree with its computed value. Either a frame has been misread (reader trouble) or a frame has been mispunched. Run to continue, but accuracy of compiled program is doubtful.**

---

* Compiler cannot be restarted. Reload MPT to run again.

** An assembly program compiled after a 7600 stop has been encountered may work for most programs assembled, as the error may be in a region of the assembly program rarely or never used by the compiler I/O configuration or in assembling certain types of programs.

PART II
OSAS-A ASSEMBLER

The 160-A Assembly System (OSAS-A) is a two or three-pass assembler depending upon the size of a symbolic program. If the condensed representation of the symbolic program exceeds available core storage, it is dumped onto some intermediate storage medium.

This intermediate information is read back in for the assembler's second pass and is read in a second time if a third pass is required. A third pass occurs only if both listable and binary output of an assembled program are to be dumped on paper tape.

During the first pass of the assembly process, the assembler reads in each line of symbolic information, scans the various fields, and stores a condensed representation of each line in a reserved block of storage. When that storage block is full, its contents are dumped onto the output medium designated by the input parameter during generation of OSAS-A. This is the intermediate output. Other steps during the first pass include assigning values to location symbols, entering symbolic quantities into the symbol table, translating operation codes, and advancing the current location counter.

During the second pass the assembler analyzes and converts the intermediate information. It transmits each line of listable output and grouping of assembled binary words to the output device. If the current symbol table will be needed during the assembly of other programs, it may be punched on a separate non-listable paper tape at this time.

Normally, OSAS-A automatically prepares the symbol table during the assembly process. However, a special symbol table may be loaded in conjunction with a symbolic program. The symbol table information to be loaded must be in the form generated by OSAS-A.

Also, intermediate output may be loaded together with the special symbol table; both must be in the form generated by OSAS-A, and they must be compatible. This facility of OSAS-A permits assembly of a segmented program or the independent assembly of one portion of a large program.

A programmer may halt the assembly process temporarily at any point by inserting a WAI pseudo-op or setting a stop, making necessary changes, and continuing. Listable or binary output may be suppressed by inserting a SUPA or SUPB pseudo-op in the program; or by positioning certain jump switches, an external symbol table may be loaded. The symbol table is normally suppressed and can be obtained as an output only by positioning a Jump switch as specified in the operation instructions.

PROGRAM RELOCATION

All programs assembled by OSAS-A are relocatable by specifying a relocation constant at load time. The relocation constant is added, under control of the loader, to the storage address assigned to relocatable words in a program assembled by OSAS-A. Relocatable words may be stored in any location in memory. Each word that is capable of being relocated must be assembled under control of the ORG-PRG counter. Therefore, to be relocatable, a word or group of words must be associated with the ORG or PRG pseudo-op as shown in the example on page 14. Words consisting entirely of addresses that refer to relocatable words in the symbolic program must be modified by the relocation constant. Words that are stored in low core locations ($0000\text{-}0077_8$) in any bank are normally non-relocatable and are assembled under control of the CON counter. Storage locations assigned to these words are not incremented by the relocation constant.

Words that will not be modified by the relocation constant are as follows:

1. Words that contain an op code or op code, address, and additive fields.
2. Words that consist of entirely numeric address and additive fields.
3. Words that refer to addresses of words assembled under control of the CON counter.

SYMBOLIC INPUT

A symbolic program to be assembled by OSAS-A may be read in from various media as designated by the input parameter from the following possibilities:

1. Paper tape punched by a Flexowriter.
2. Magnetic tape from CONTROL DATA 163 Tape Units.
3. Magnetic tape (low density) from IBM 729 tape units via CONTROL DATA 1608 Adapter.
4. Magnetic tape from CONTROL DATA 1607 Tape Units.

5. Punched cards from a fast card reader via CONTROL DATA 1610 Control Unit.
6. Punched cards from the CONTROL DATA 167-1, 167-2, or 405 Card Reader.

Acceptable BCD or Flexowriter input characters and corresponding output characters are shown in table A, page 19. All input is converted to BCD in the computer. Remarks and other pseudo-ops may be located anywhere in a program; however, each program must end with a WAI or an END pseudo-op.

The location, address, and additive fields in any input medium are preceded by a plus, a minus, or a blank. (A blank indicates a positive field.) The description of each field in a line of symbolic input is given below.

1. Location Field

   A location symbol may be a maximum of six alphanumeric characters. The sign (or blank) is not a part of the location field. Numeric characters may be either octal or decimal. Resultant octal values equal to or greater than $10{,}000_8$ may be used as location symbols. Values less than $10{,}000_8$ will be entered in the symbol table but can not be referenced by other instructions. Four-digit decimal numbers that result in five-digit octal values are valid symbols.

2. Operation Code Field

   This field may contain any of the symbolic op codes or pseudo-ops listed in appendix A.

   Two-digit octal machine instruction codes may also be used if they are left-justified within the field.

3. Address Field

   This field may contain an octal number, a decimal number followed by a D, or a location symbol. Any location symbol used in an address field must appear in the location field somewhere in the program.

4. Additive Field

   This field increments the address specified in the address field. The field may contain a location symbol, an octal integer, or a decimal number followed by a D.

   Information specified by the additive field will be added algebraically to information specified in the address field. Location symbols appearing in either field are represented by assigned machine addresses during address computation.

In the relative mode, the resultant sum is added to or subtracted from the current address to obtain a new address. The resultant sum can never exceed $\pm 77_8$ except for Memory Address Mode instructions, the Return Jump instruction, or the Selective Jump instructions. If a minus sign is punched in the sign position preceding the location field of a line, the current address is ignored in processing the address and additive fields for that line.

5. Comments Field

   This field may contain up to 50_{10} characters. BCD and flex character inputs are translated for output as a function of the I/O device used. Formats are shown in table A on page 19.

CARD SYMBOLIC FORMAT

Columns	Contents	Columns	Contents
2	Minus, plus sign, or blank	16-21	Address Field
3-8	Location symbol	23	Minus, plus sign, or blank
10-13	Op Code	24-29	Additive Field
15	Minus, plus sign, or blank	31-80	Comments Field

Contents of the location, address, and additive fields may be left-justified (may start in columns 2, 15, 23).

Arrangement of Symbolic Deck for Assembly

No special start or identification card is required. A remarks card may be included to identify the program but is not necessary. Blank cards are ignored; they do not affect the object program. The last card must be an END or a WAI card. Decks may be followed by 3 blank cards to assure reading of the last card.

Decks may be stacked for assembly by placing several programs in the reader, one program following another with no special cards between.

SYMBOLIC PAPER TAPE FORMAT

Paper tape input is punched on seven level tape by a Flexowriter. Every tape must begin with a carriage return and each line must terminate with a carriage return. Lines may be of variable length; location, address, and additive fields (including the sign) may also vary within the specified limits. Each field is terminated by a tab, and each tape must have at least one inch of trailer. A line may be terminated at the end of any field by typing a carriage return instead of a tab.

A program may be continued to a second tape by placing a WAI pseudo-op at the end of the first tape so that the program will pause while a second tape is placed in reading position. Each tape must begin with a carriage return and all except the last one must end with a WAI. The last tape must end with an END pseudo-op.

Tape Format: SIGN LOC TAB OP TAB SIGN ADDRESS TAB SIGN ADDITIVE TAB COMMENTS CR

SYMBOLIC MAGNETIC TAPE FORMAT

A card to tape routine is currently not included with OSAS-A. The user may write a card-to-tape routine or use an off line card-to-tape device. OSAS-A expects to find symbolic magnetic tape input written in binary coded decimal (BCD) with card images in the format shown under SYMBOLIC CARD INPUT on page 10. Blank cards need not follow a deck of cards for card-to-tape. If programs are stacked for assembly, the operator must know how many are on a tape since there is no indicator to mark the end of the last program.

OSAS-A PSEUDO-INSTRUCTIONS

The OSAS-A assembly program recognizes 16 pseudo-instructions (pseudo-ops). Pseudo-ops are not interpreted as machine language instructions; they provide the programmer with a means of controlling the assembly of a symbolic program, and are recognized only by the assembly program. The ORG, PRG and CON pseudo-ops are included in the pseudo-op repertoire because of the relative addressing feature of the 160-A computer. These three pseudo-ops determine the storage location of each assembled line in the object program by specifying the contents of the two assembly program location counters - the ORG-PRG counter and the CON counter. Addresses assigned to each line during assembly are under control of whichever counter is active at that time. The active location counter is incremented after the current line is assembled. The CON counter is actuated by a CON pseudo-op and the ORG-PRG counter is actuated by either an ORG or a PRG pseudo-op. At the beginning of a program, OSAS-A automatically sets the ORG-PRG counter to 100_8 and the CON counter to 0. Relocatable lines are assembled under control of the ORG-PRG counter; non-relocatable lines (those that are to be placed in low core) are assembled under control of the CON counter.

ORG-PRG Counter

This counter is activated by either an ORG or a PRG pseudo-op. The counter is set to the value of the algebraic sum of the address field and the additive field EXCEPT when both these fields are blank. The instruction immediately following an ORG or a PRG pseudo-op is assembled to the location contained in the ORG-PRG counter as a result of the pseudo-op. If neither of the pseudo-ops appear at the beginning of a program, the ORG-PRG counter is set to 0100.

If the additive and address (AA) fields are blank the following occurs:

1. ORG causes the ORG-PRG counter to be set to 100_8.
2. PRG causes the ORG-PRG counter to resume control and to continue counting from the previous value.

CON Counter

This counter is initially set to 0 if no CON pseudo-op occurs at the beginning of a program. When a CON pseudo-op is encountered, the CON counter is set to the algebraic sum of the address and additive (AA) fields. This sum must never be greater than 77_8. As long as the CON counter remains active, it is incremented by one except when it is reset by the AA fields of a CON pseudo-op. If the AA fields of a CON pseudo-op are blank, counting begins from the previous value. The legal range of a CON counter is from 0 through 77_8. If locations higher than 77_8 are specified by a CON pseudo-op, they are flagged as range errors.

Symbol Table

Each symbol in the location field of a line in a symbolic program is assigned a numeric value by OSAS-A and is placed in a symbol table. This symbol table has two parts, variable and constant. The variable symbol table contains the numeric value of all symbols that refer to relocatable words. The constant symbol table contains the numeric value of all non-relocatable symbols. Symbols that are assigned to constants or low core addresses are placed in the constant symbol table. The capacity of the entire symbol table is 1000_{10}. The following formula may be used for computing the allowable variable or constant symbol table capacity in a program to be assembled.

$$S_v + 2S_c = 1000_{10}$$

Where S_v = total number of variable symbols

and S_c = total number of constant symbols

External symbol tables may be loaded with a symbolic program at assembly time if the external symbol table is in a format acceptable to OSAS-A.

Pseudo-Ops

ORG - Puts the ORG-PRG counter in control and causes the ORG-PRG location counter to assume the value of the algebraic sum of the address and additive (AA) fields. This value is the location to which the next instruction (after an ORG) will be assembled. Each word assembled under this pseudo-op will be flagged as relocatable on the binary output card or card image. Symbols appearing in either AA field must be defined before ORG is executed. An ORG should not be used to continue an assembly. Blank AA fields set the ORG-PRG counter to 0.

PRG - Has all the properties of ORG but it is used principally to continue an assembly. A PRG with blank AA fields reactivates the ORG-PRG counter. If a PRG with blank AA fields is the first instruction of a program, the counter starts at 100_8.

CON - Controls the CON counter as the PRG controls the ORG-PRG counter. Symbolic locations defined under control of the CON pseudo-op may be referenced with no-address, direct, or indirect address instructions only. Words assembled under control of the CON pseudo-op are non-relocatable, and may be stored only in low core.

NOTE: Forward, backward and relative mode instructions must not refer to symbolic locations defined by the CON pseudo-op. If such references are made, errors that are extremely difficult to interpret may appear in the assembled listing.

EXAMPLE

LOC	SYMBOL	OP	ADDRESS	ADDITIVE
		ORG	1000	
*1000	OMEGA	LDD	ALPHA	
*1001		STD	BETA	
*1002		AOD	DELTA	
*1003		RAD	DELTA	+1
		CON	43	
0043	ALPHA		35	
0044	BETA			
0045	DELTA			
		PRG	(ORG with blank AA fields would set LDF location to 0. ORG with AA fields OMEGA + 4 would set LDF location to 1004.)	
*1004		LDF	**THETA	
*1005 + 1006		STM	**STORE	
*1007 + 1010		JPR	**OMEGA	
*1011	THETA			
*1012	STORE			
		END		

* Will be incremented by a relocation constant
** Will be modified by a relocation constant

The remaining pseudo-ops recognized by the OSAS-A Assembly System are as follows:

BLR, BSS	Advances the location counter in control by the amount specified in the address plus additive field. Any symbol in the location field will be assigned to the first numeric address in the block. Care must be taken not to exceed the CON counter range.
WAI	Stops the assembly program to allow for mounting a new input tape. The END pseudo-op may be simulated here by entering a non-zero quantity into the A register and running.
END	Prepares the assembly program for the second pass. During binary output, a transfer card is produced. An END pseudo-op with blank AA fields will cause the loader to transfer program control to address 0000. Otherwise an END pseudo-op will transfer program control to the address designated by the algebraic sum of the AA fields. NOTE: An END or WAI pseudo-op must terminate the assembly program. If a WAI is the last instruction, the END pseudo-op must be generated manually in order to complete the assembly process.
EQU	Assigns the algebraic sum of the address and additive fields of the EQU pseudo-op to the symbol in the location field, and places this symbol and its numeric value in either the variable or the constant symbol table. The address and additive fields may be either numeric or symbolic. If both fields are numeric, the symbol in the location field is assigned to the constant symbol table and is not relocatable. (To maintain program relocatability, numeric locations should refer only to low core.) If both address and additive fields are symbolic, the algebraic sum of the two fields are equated to the symbol in the location field. If either address or additive symbol is relocatable, the symbol in the location field will be relocatable. If both symbols are non-relocatable, the location field symbol will be non-relocatable (constants or low-core addresses). If the address and additive fields consist of a symbol and a numeric value, the relocatability of the location field symbol is determined by the relocatability of the symbol in either field.
REM	Treats as remarks all that follows the additive field and is ignored by the assembly program. A REM instruction will not cause the location counter to advance. A maximum of 50_{10} characters is permitted in the remarks field.

BNKX	Generates a binary bank card that will cause the instructions following the pseudo-op to be loaded into bank X. (X must be an octal digit.) This pseudo-op does not affect the assembly of a symbolic program; it is a signal to the loader.
SUPA	Suppresses the assembly listable output.
SUPB	Suppresses the binary output.
BCD	Causes a contiguous string of characters to be assembled 2 BCD characters per word.
BCDR	Causes a contiguous string of characters to be assembled 1 BCD character per word.
FLX	Causes a contiguous string of characters to be assembled 2 flex characters per word.
FLXR	Causes a contiguous string of characters to be assembled 1 flex character per word.

The following rules affect BCD, BCDR, FLX and FLXR pseudo-ops. A maximum of 50_{10} characters is permitted by the BCD, BCDR, FLX, and FLXR pseudo instructions. The characters must be in the remarks field; a character count must appear in the address field.

ASSEMBLER RULES FOR INPUT

1. A line containing information in the remarks field only will be ignored during assembly but will appear in proper order with the listable output.
2. All non-printing characters (color shift) are ignored by the assembler except in the case of control characters (carriage return).
3. If the OP field of a line is blank, the address and additive fields are added algebraically and stored as one 12-bit number. (If the result is negative, the line will assemble without error, but the resultant machine word will be incorrectly modified by any relocation constant.)
4. If the OP field is non-relative, the address and additive fields are added algebraically to form a 6-bit low-core machine address or a 6-bit constant. If the sum of the address and additive field exceeds 77_8, the line is flagged as a range error in the listable output and the low order 6-bits of the instruction are set to zero.

5. If the OP field is relative (last character is F, B, or R), the contents of the location counter are subtracted from the algebraic sum of the address and additive fields. (A minus sign in the sign position of the location field indicates that a line is to be assembled as in rule 2.) For F-type op codes, a positive result is directly inserted in the low order 6-bits of the resultant 12-bit machine instruction. If the result is negative, the low order 6-bits are cleared to zero and the line is flagged as a possible range error in the listable output. For B-type op codes, a negative result is complemented before it is combined with the high order 6-bit machine op code. If the result is positive, the low order 6-bits are cleared and the line flagged as a possible range error.

   The possibility of range errors may be reduced by substituting for F or B-type op codes an R-type op code, which is a pseudo-op recognized by OSAS-A. If, for example, a programmer wishes to use one or the other of the two relative machine codes (F or B), but does not know at the time the program is written whether the desired reference is forward or backward, an R-type code may be used. An R-type (special relative) op code forces OSAS-A to examine the result of the subtraction of the contents of the location counter from the sum of the address and additive fields to determine the correct relative machine op code. If the result is positive, the machine op code will be an F-type; if the result is negative, the machine op code will be a B-type. (JPR and ERR are not considered R-type op codes and will not be recognized as such by OSAS-A.) For all three types of relative op codes, if the difference between the current location and the sum of the address and additive fields is greater than $\pm 77_8$, the low order 6-bits of the resulting machine instruction are cleared to zero and the line is listed with a range error flag.

LISTABLE OUTPUT

Listable output may be on any medium specified by the input parameter. Flexowriter or printer listings can list 50 characters in the comment field. Listing from cards limits comments to the first 32 characters.

Columns	Contents
2-5	Error flags
7-10	Octal location (location in core to which the line of data was assembled)
13-16	Octal contents (contents of octal location)

Columns	Contents
19	Blank or minus sign
20-25	Location symbol
27-30	Op code
32	Plus sign, minus sign, or blank
33-38	Address
40	Plus sign, minus sign, or blank
41-46	Additive
48-97	Comments

Error Flags

OSAS-A has eight error codes which may appear on an assembly listing.

Error Code	Explanation
c	CON location out of range (beyond first 100_8 locations)
e	E term of assembled machine op code greater than 77_8
l	Undefined symbol in location field
m	Symbol defined more than once
o	Illegal op code
u	Undefined symbol in address field
v	Undefined symbol in additive field
x	Illegal character

TABLE A. RULES FOR CHARACTER CORRESPONDENCE

FLEX INPUT

Input	Output	Output
FLEXOWRITER CHARACTER	FLX, FLXR	BCD, BCDR
	PART I (one to one character correspondence)	
A thru Z	A thru Z	A thru Z
0 thru 9	0 thru 9	0 thru 9
-	-	- (minus)
,	,	,
/	/	/
.	.	.
)	)	)
Blank	Blank	Blank
'	'	*
=	=	=
(	(	(
:	:	$
+	+	+
	PART II (special characters)	
color shift	upper case code	plus zero
double color shift	color shift	double plus zero
	PART III (control character combinations)	
, ; (to terminate a field)	tab (terminates field)	, $
, . (to terminate a line)	carriage return (terminates line)	, .

TABLE A (CONT'D)

BCD INPUT

Input	Output	Output
BCD CHARACTER	FLX, FLXR	BCD, BCDR
	PART I (one to one character correspondence)	
A thru Z	A thru Z	A thru Z
0 thru 9	0 thru 9	0 thru 9
- (minus)	-	-
,	,	,
/	/	/
.	.	.
)	)	)
Blank	Blank	Blank
*	'	*
=	=	=
(	(	(
$	:	$
+	+	+

20 Insert the following conversions after the + characters.

	PART II (special characters)	
plus zero	upper case code	plus zero
double plus zero	color shift	double plus zero
	PART III (control character combinations)	
, $	tab (terminates field)	, $
, .	carriage return (terminates line)	, .

BINARY OUTPUT

All binary output produced by OSAS-A is relocatable; it includes program cards, bank cards, and a transfer card for each program.

Magnetic tape binary output consists of complete binary card image records, 160 characters per record.

The paper tape binary output routine suppresses unused words of the card image. Each new card image on paper tape is indicated by a seventh level punch in column 1. Format is given below:

Columns	Rows	Information
Program Card		
1	7, 9, 12	Binary program card indicators
1	8	If check sum is to be ignored
1	0-6	Word Count
2		Starting address
3		Check sum of other 79 columns
4-9		Designator bits for words in columns 10 through 80 that must be modified by relocation constant. A punch indicates that word will be modified at load time if relocation constant is specified. A punch in row 9, column 9, indicates that words in columns 10 through 80 will not be relocated. (A 12-punch in column 4 indicates that word 10 is modified; an 11-punch in column 4 indicates that word 11 is modified; a 0-punch in column 4 indicates word 12 is modified. Ordering of word-bit correspondence proceeds in sequence down each column, 4 through 9.)
10-80		Machine instructions, addresses or constants.
Bank Card		
1	7, 9, 11	Bank card indicator
1	8	If check sum is to be ignored
2	7-9	Bank number
3		Check sum
Transfer Card		
1	7, 9, 12	Binary transfer card indicator
1	8	If check sum is to be ignored
2		Transfer address
3		Check sum

OPERATING INSTRUCTIONS FOR ASSEMBLY PROCESS

Magnetic Tapes

Tape assignments are:	Unit #1 - Symbolic input
	Unit #2 - Intermediate input/output
	Unit #3 - Listable output
	Unit #4 - Binary output

To rewind unload tapes 1, 3 and 4 after final halt, run - DO NOT CLEAR. Tape 1 will not rewind if Selective Jump switch 1 is set.

To Assemble

a) Normal Assembly - no intermediate I/O
   1. Master clear
   2. Set bank control registers to zero
   3. Run
   4. At stop 2152, program has been assembled correctly. Run to rewind unload tape units or master clear and run to assemble next program. If stop 2151 occurs, certain lines of symbolic input were in error. The A register displays number of lines in error. Run to stop 2152.

b) Normal Assembly - with intermediate I/O
   1. to 3. Same as a)
   4. At stop 1152, load intermediate I/O for cards or paper tape (magnetic tape reads back in automatically) by positioning and running.
   5. Same as 4. in a)

c) Assemble with given symbol table - no intermediate I/O
   1. Position symbol table in reader, set Jump switch 2
   2. to 4. Same as steps 1-3 in a)
   5. At stop 0117, position symbolic program for loading, and run.
   6. Same as 4. in a)

d) Assemble with given symbol table and intermediate I/O.
   1. Position symbol table in reader. If intermediate I/O is not on paper tape, position it for reading at the same time.
   2. Enter 1051 into P register and run.

3. At halt 1152, position intermediate I/O and run.

   NOTE: The assembly will halt at 1152 only if intermediate I/O is on paper tape or cards. Magnetic tape will be read in automatically with no stop.

e) Punch symbol table during assembly.

   1. Prior to assembling for any option, set Jump switch 4.
   2. Proceed with assembly.

f) Halt program after first pass with magnetic tape unit 1 in rewind load position.

   1. Prior to running assembly program, set Jump switch 1.
   2. Proceed with assembly.
   3. At stop 1042, first pass is complete and tape unit 1 is in rewind load position. Run from this point unless symbolic input tape is to be saved.

g) Halt assembly program prior to completion.

   1. Set Stop switch 1. Do NOT take out of run position to stop assembly program.
   2. To resume assembly, restore Stop switch 1 and run.
   3. To restart assembly program, master clear and run.

   NOTE: If OSAS-A is being loaded from a previously prepared bi-octal tape, always load it first with P and A registers at zero before starting assembly procedure.

Stops

Normal Stops:

1042	Will occur only if Jump switch 1 has been set. Occurs after the first pass so that magnetic tapes and tape designations can be changed in a two-tape I/O system.
0117	Symbol table has been read. Position symbolic input and run to begin assembly with a given symbol table.
1152	Position intermediate I/O for input and run.
2152	Final stop. The assembly has been completed correctly. Run if tape units are to be rewound unload. Master clear and run to assemble next program.

2200 Post-final stop occurs after running from stop 2152. Master clear and run to assemble next program.

2277 The WAI pseudo-op has been encountered. Position next symbolic input portion and run to continue assembly. To simulate an END pseudo-op, enter some non-zero quantity into the A register and run.

Error Stops:

2151 Line error display. The number of program lines containing errors is displayed in the A register. Run to proceed to final stop (2152).

0343 or 3763 Symbol table is too large for computer to handle. Either the number of symbols must be decreased, or the program must be assembled in parts; symbols common to more than one part must be EQU'd to each other. Each part (except the first) must be ORG'd to the address immediately after the last address of the preceding portion. Maximum symbol table capacity is 1000_{10}.

Additional Error Stops (as a function of I/O medium):

Some error stops are computed as a function of I/O subroutine length. The following tables should be used to calculate the various error stops. When a halt occurs at a parity error, the program has attempted to read or write three times.

Second Record Subroutine Lengths (in octal)

Main Driver	L = 4412

Symbolic Input

Paper Tape	L = 141
Magnetic Tape (163)	L = 57
1610 Card Reader (Fast)	L = 274
167 Card Reader	L = 124
Magnetic Tape (1608 Low Density)	L = 104
Magnetic Tape (1607)	L = 107
405 Card Reader	L = 76
167-2 Card Reader	L = 72

Intermediate I/O

Paper Tape	L = 201
Magnetic Tape (163 Assembly Mode)	L = 105
Magnetic Tape (163 Character Mode)	L = 163
Magnetic Tape (1608 Low Density)	L = 136
Magnetic Tape (1607)	L = 122

Listable Output

Paper Tape	L = 147
Magnetic Tape (163)	L = 43
Line Printer (1612)	L = 23
Line Printer (166)	L = 66
Magnetic Tape (1608 Low Density)	L = 116
Magnetic Tape (1607)	L = 75

Binary Output

Paper Tape	L = 122
Magnetic Tape (163 Character Mode)	L = 72
1610 Card Punch (Fast)	L = 122
Magnetic Tape (1608 Low Density)	L = 105
Magnetic Tape (1607)	L = 66

Error Stops (in octal)

Magnetic Tape (163)

1. Symbolic Input EOF mark detected; run to generate an END card.	Contents of location 62 + 33
2. Intermediate (Character Mode) Write parity error; run to rewrite. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 55
Read parity error; run to reread. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 151

3.	Intermediate (Assembly Mode)	
	Write parity error; run to rewrite. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 30
	Read parity error; run to reread. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 77
4.	Listable Output	
	Write parity error; run to rewrite. Zero A register and run to disable parity for this record. Continue processing.	Contents of location 65 + 26
5.	Binary Output (Character Mode)	
	Write parity error; run to rewrite. Zero A register and run to disable parity for this record. Continue processing.	Contents of location 66 + 62

<u>Magnetic Tape (1608)</u>

1.	Symbolic Input	
	End-of-file sensed	Contents of location 62 + 33
2.	Intermediate I/O	
	Write parity error; run to rewrite. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 31
	Read parity error; run to reread. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 115
3.	Listable Output	
	Write parity error; run to rewrite. Zero A register and run to disable parity for this record. Continue processing.	Contents of location 65 + 56
4.	Binary Output	
	Write parity error; run to rewrite. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 66 + 34

Magnetic Tape (1607)

1.	Symbolic Input	
	End-of-file marks sensed	Contents of location 62 + 35
	Read parity error indicated by rapid back-and-forth tape movement. (OSAS-A is attempting to read record correctly.)	
2.	Intermediate I/O	
	Write parity error; run to rewrite. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 27
	Read parity error. Run to reread. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 63 + 101
3.	Listable Output	
	Write parity error; run to reread. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 65 + 52
4.	Binary Output	
	Write parity error; run to reread. Zero A register and run to disable parity for remainder of record. Continue processing.	Contents of location 66 + 31

Paper Tape

1.	Symbolic Input	No Stop
2.	Intermediate I/O	
	Frame just read has lateral parity error. Step twice, enter correct character in A register and run.	Contents of location 62 + 140
	Frame just read in is not longitudinal check character; re-start assembly.	Contents of location 63 + 175
3.	Listable Output	No Stop
4.	Binary Output	No Stop

Cards (1610 Fast)

1.	Symbolic Input Card Reader Error	Contents of location 62 + 20
2.	Binary Output Equipment not ready.	Contents of location 66 + 113

Cards (167-1)

1.	Symbolic Input 167-1 card reader not operating. A register displays 33XX. Correct 167 error condition and master clear. Add one to contents of P, enter sum in P register, zero A register and run.	Contents of location 62 + 121 4324 + 117

167 Status Response Codes	XX = 00 Card read ready
	XX = 01 Hopper empty
	XX = 02 Stacker full
	XX = 04 Feed failure
	XX = 10 Program error
	XX = 20 Amplifier failure
	XX = 40 Motor power off

Cards (167-2)

1.	Symbolic Input 167-2 card reader not operating. A register displays 33XX. Correct 167-2 error condition and master clear. Add 1 to contents of P, enter sum in P register, zero A register, and run.	Contents of location 62 + 67

Cards (405)

1.	Symbolic Input 405 card reader not operating. A register displays 3XXX. Correct 405 error condition and master clear. Add 1 to contents of P, enter sum in P register and run.	Contents of location 62 + 71

405 Status Response Codes	XXX = 000 Card reader ready
	XXX = 001 Hopper empty
	XXX = 002 Stacker empty
	XXX = 004 Feed failure
	XXX = 010 Program error
	XXX = 020 Amplifier error
	XXX = 040 Motor power off
	XXX = 100 Compare error
	XXX = 200 Hopper empty with EOF set

Printer 166

1. Listable Output

Printer in off-line condition. Place printer on-line and run.	Contents of location 65 + 13
Printer is out of paper.	Contents of location 65 + 16

MNEMONIC CODES FOR OSAS-A

The 160-A Assembly System recognizes all of the normal mnemonic operation codes for the 160-A computer. It also recognizes certain pseudo-ops* and 15 special relative codes. The special relative codes are relative instructions which do not specify direction. An R is substituted for the terminal B or F in the symbolic program and OSAS-A determines the proper direction automatically as explained on page 17. For example, LPR may be coded instead of LPF or LPB. The special relative mnemonics are: ADR, AOR, LCR, LDR, LPR, LSR, NJR, NZR, PJR, RAR, SBR, SCR, SRR, STR, ZJR.

Appendix A contains a complete list of all mnemonic codes recognized by OSAS-A, including normal operation codes, pseudo-ops, and the special relative codes. To provide compatability with previous 160 assembly programs, OSAS-A accepts the old Shift A and Logical Sum instructions.

* Discussed on page 12.

PART III

MASTER PAPER TAPE EDIT

This program will produce on OSAS, OSAS-A, OSAS-AX or AUTOCOMM Master Paper Tape. It allows replacement of the compiler (bi-octal) portion and the driver portion of the MPT. The I/O routines can be replaced, deleted, or a new routine inserted.

If the compiler is to be replaced, the new compiler tape must be in bi-octal format. All other routines are in OSAS or OSAS-A binary format.

EDIT PARAMETERS

The edit parameters are entered in the computer by manually placing them in the A register. If the compiler is to be replaced, its parameter must be entered first. The next parameter which must be entered is the driver's, if it is to be replaced. The remainder of the parameters may be entered in any order. The last parameter must be 0077.

PARAMETERS

0000	replace compiler
0001	replace driver
01XX	delete routine XX (ID number)
00XX	replace or insert XX (ID number)
0077	end of list

ID numbers of routines are assigned as follows:

10	symbolic input $\leq$ 17
20	intermediate output $\leq$ 27
30	listable output $\leq$ 37
40	binary output $\leq$ 47

OPERATING INSTRUCTIONS

To Edit a New MPT

1. Position edit tape in reader; master clear and RUN.
2. At halt P = 650, A = 5043, set the Load/Clear switch to Clear and the Run/Step switch to Neutral.
3. Set P = 0000. If the compiler is to be replaced, place new compiler routine (bi-octal format) in reader, otherwise place old MPT in reader, RUN.
4. At stop P = 60, enter edit parameter in A register. Place Run/Step switch to Run and repeat until all parameters have been entered. After last edit parameter has been entered, place 0077 in A register and RUN.
5. If compiler has been replaced and driver will not be replaced, a stop with P = 0121 occurs. Replace MPT in reader, positioned at blank space following the old compiler. Set Run/Step switch to Run.
6. If the driver is to be replaced a halt with P = 0143 occurs. Place the new driver tape in reader and set Run/Step switch to Run. After the new driver has been reproduced, a halt at P = 0175 will occur. Place the MPT in reader at the blank following the bi-octal portion of the tape, set the Run/Step switch to Run.
7. If an I/O routine is to be replaced or inserted, a halt at P = 0330 will occur with the ID number of the routine to be replaced in the A register. Place the routine in the reader, and set the Run/Step switch to Run. Repeat this procedure until all routines have been punched.
8. If at the completion of an edit a routine was specified to be deleted but was not on the MPT, a halt at P = 0444 occurs with the specified routine ID numbers in the A register. Set Run/Step switch to Run position to determine if any other delete errors occurred.
9. At the completion of the edit, a stop will occur at P = 0446.

To Copy a MPT

1. Load edit program tape at 0000. Halt P = 0650, A = 5043 indicates program tape is properly loaded.
2. Place MPT to be copied in reader. Set P = 0002 and RUN. At halt P = 0446, MPT has been copied.

To Determine Routines on a MPT

1. Load edit program tape at 0000. Halt P = 0650, A = 5043 indicates program tape is properly loaded.
2. Position MPT in reader.
3. Set P = 0004 and RUN.
4. Halt with P = 0050 occurs with the ID of next routine on MPT in A register. Run until 77 appears in the A register signifying the end of the MPT.

Verifying the New MPT

Test all possible I/O configurations to insure that the new MPT will compile properly.

PROGRAM STOPS FOR MASTER PAPER

Normal Stops

0050	Halt with ID in A register. Used to determine routines on MPT.
0060	Halt to enter Edit parameter in A register.
0121	Halt to replace MPT in reader after replacing compiler.
0143	Halt to enter new driver tape in reader.
0175	Halt to replace MPT in reader after replacing driver routine.
0330	Halt with ID number of routine to be replaced or inserted in A register.
0446	Final halt for copy and edit.

Error Stops

0444	Halt to indicate a routine which was to be deleted was not on the MPT.

PART IV
BINARY LOADERS

The OSAS-A assembly system includes a loading routine for each form of binary input. Loaders are in relocatable format stored on bi-octal paper tapes (not on the MPT); they load binary card images from cards, magnetic tape, or paper tape prepared by the OSAS-A assembly program.

OPERATING INSTRUCTIONS

1. Place bi-octal loader tape in reader, enter starting address into P register, load and run.
2. At halt, set P register to starting load address, position first binary input (tape unit #4 if magnetic tape), enter relocation constant of first binary input into A register and run.
3. To load additional binary inputs:
   Enter original load address into P register, position next binary input, enter its relocation constant into A register and run.

   NOTE: An assembled program also may be relocated during input by placing a relocation card before the binary deck. In such cases the relocation constant need not be entered into the A register. The relocation card format is:

Columns	Rows
1	7, 9, 11 and 12 punches
	8 if check sum is to be ignored
2	relocation constant
3	check sum

STOPS

Loader stops are computed as a function of the input medium and the length of the binary loader. Use the following tables to compute the proper stop addresses.

Binary Loader Lengths
(in octal)

163 Magnetic Tape, Assembly Mode	L = 412
163 Magnetic Tape, Character Mode	L = 555
1607 Magnetic Tape	L = 412
1608 Magnetic Tape	L = 411
1610 Card Reader, Fast	L = 613
Paper Tape	L = 425
167-1 Card Reader	L = 366
405 Card Reader	L = 402

NORMAL STOPS

(L = first storage location of loader)

Cards			Magnetic Tape		Paper Tape	Magnetic Tape (163)		
405	1610	167	1607	1608		Assy. Mode	Char. Mode	
L + 227	L + 440	L + 213	L + 216	L + 215	L + 253	L + 216	L + 361	- Transfer card has been read. Run to cause a jump to program and execute. Master clear, enter starting load address into P register, enter relocation constant into A register, and run to load next program. If magnetic tape loaders are used, to rewind unload tape unit #4, zero the A register and run from the halt.
			L + 222	L + 221		L + 222	L + 365	- Tape unit #4 has been rewound unload. Run from here to jump to program. To load next program, change tapes, master clear, enter starting load address into P register, relocation constant into A register and run.
			L + 144	L + 143		L + 143	L + 270	- End of file mark detected on last read operation; run to rewind unload magnetic tape unit #4; clear A register and run to read next record.
			L + 150	L + 147		L + 147	L + 274	- Tape unit #4 has been rewound unload.

ERROR STOPS

Cards			Magnetic Tape		Paper Tape	Magnetic Tape (163)		
405	1610	167	1607	1608		Assy. Mode	Char. Mode	
			L + 160	L + 157	L + 140	L + 160	L + 305	- Read parity error. Run to continue processing.
L + 174	L + 405	L + 160	L + 200	L + 177	L + 220	L + 200	L + 343	- Card or record just read not in proper binary format. Run to continue processing.
L + 226	L + 437	L + 212	L + 215	L + 214	L + 252	L + 215	L + 360	- Check sum error in transfer card; step once and run to jump to program.
L + 344	L + 555	L + 330	L + 355	L + 354	L + 367	L + 355	L + 520	- Check sum error. Run to read next record or card.
L + 147								- Reader failure.

APPENDIX A
MNEMONIC CODES RECOGNIZED BY OSAS-A

X or XX = Any Octal digit or digits
XXXX = Octal Operand or EF code
YYYY = Octal Address

Mnemonic	Name	F	E	G	Type*
ACJX	Set D, I, and R Bank Control and Jump	00	7X		N
ADB	Add Backward	33	XX		N
ADC	Add Constant	32	00	XXXX	N
ADD	Add Direct	30	XX		N
ADF	Add Forward	32	XX		N
ADI	Add Indirect	31	XX		
ADM	Add Memory	31	00	YYYY	N
ADN	Add No Address	06	XX		N
ADR	Add Relative	3X	XX		S
ADS	Add Specific	33	00		N
AOB	Replace Add One Backward	57	XX		N
AOC	Replace Add One Constant	56	00	XXXX	N
AOD	Replace Add One Direct	54	XX		N
AOF	Replace Add One Forward	56	XX		N
AOI	Replace Add One Indirect	55	XX		N
AOM	Replace Add One Memory	55	00	YYYY	N
AOR	Replace Add One Relative	5X	XX		S
AOS	Replace Add One Specific	57	00		N
ATE	A to Buffer Entrance Register	01	05	YYYY	N
ATX	A to Buffer Exit Register	01	06	YYYY	N
BCD	2 BCD Characters/Word				P
BCDR	1 BCD Character/Word				P
BLR	Block Storage Reserve				P
BLS	Block Store	01	00	YYYY	N
BNKX	Generate Binary Bank Card				P

* Normal codes = N, pseudo-ops = P, and special relative codes = S

Mnemonic	Name	F	E	G	Type*
BSS	Block Storage Reserve				P
CBC	Clear Buffer Controls	01	04		N
CIL	Clear Interrupt Lockout	01	20		N
CON	Constant				P
CTA	Bank Controls to A	01	30		N
DRJX	Set D and R Bank Control and Jump	00	5X		N
END	End				P
ETA	Buffer Entrance Register to A	01	07		N
EQU	Equality				P
ERR	Error Stop	00	00		N
EXC	External Function Constant	75	00	XXXX	N
EXF	External Function Forward	75	XX		N
FLX	2 Flex Characters/Word				P
FLXR	1 Flex Character/Word				P
HLT	Halt	77 77	00 77		N
HWI	Half Write Indirect	76	XX		N
IBI	Initiate Buffer Input	72	00	YYYY	N
IBO	Initiate Buffer Output	73	00	YYYY	N
INA	Input to A	76	00		N
INP	Input	72	XX	YYYY	N
IRJX	Set I and R Bank Control and Jump	00	3X		N
JFI	Jump Forward Indirect	71	XX		N
JPI	Jump Indirect	70	XX		N
JPR	Return Jump	71	00 •	YYYY	N
LCB	Load Complement Backward	27	XX		N
LCC	Load Complement Constant	26	00	XXXX	N
LCD	Load Complement Direct	24	XX		N
LCF	Load Complement Forward	26	XX		N
LCI	Load Complement Indirect	25	XX		N
LCM	Load Complement Memory	25	00	YYYY	N
LCN	Load Complement No Address	05	XX		N

* Normal Codes = N, pseudo-ops = P, and special relative codes = S

Mnemonic	Name	F	E	G	Type*
LCR	Load Complement Relative	2X	XX		S
LCS	Load Complement Specific	27	00		N
LDB	Load Backward	23	XX		N
LDC	Load Constant	22	00	XXXX	N
LDD	Load Direct	20	XX		N
LDF	Load Forward	22	XX		N
LDI	Load Indirect	21	XX		N
LDM	Load Memory	21	00	YYYY	N
LDN	Load No Address	04	XX		N
LDR	Load Relative	2X	XX		S
LDS	Load Specific	23	00		N
LPB	Logical Product Backward	13	XX		N
LPC	Logical Product Constant	12	00	XXXX	N
LPD	Logical Product Direct	10	XX		N
LPF	Logical Product Forward	12	XX		N
LPI	Logical Product Indirect	11	XX		N
LPM	Logical Product Memory	11	00	YYYY	N
LPN	Logical Product No Address	02	XX		N
LPR	Logical Product Relative	1X	XX		S
LPS	Logical Product Specific	13	00		N
LSB	Logical Sum (exclusive or)	17			N
LSD	Logical Sum (exclusive or)	14			N
LSF	Logical Sum (exclusive or)	16			N
LSI	Logical Sum (exclusive or)	15			N
LSN	Logical Sum (exclusive or)	03			N
LSR	Logical Sum (exclusive or)	1X			S
LS1	Left Shift One	01	02		N
LS2	Left Shift Two	01	03		N
LS3	Left Shift Three	01	10		N
LS6	Left Shift Six	01	11		N
MUH	Multiply A by One Hundred	01	13		N
MUT	Multiply A by Ten	01	12		N

* Normal codes = N, pseudo-ops = P, and special relative codes = S

Mnemonic	Name	F	E	G	Type*
NJB	Negative Jump Backward	67	XX		N
NJF	Negative Jump Forward	63	XX		N
NJR	Negative Jump Relative	6X	XX		S
NOPX**	No Operation	00	0X		N
NZB	Non-Zero Jump Backward	65	XX		N
NZF	Non-Zero Jump Forward	61	XX		N
NZR	Non-Zero Jump Relative	6X	XX		S
OTA	Output from A	76	77		N
OTN	Output No Address	74	XX		N
OUT	Output	73	XX	YYYY	N
ORG	Origin				P
PJB	Positive Jump Backward	66	XX		N
PJF	Positive Jump Forward	62	XX		N
PJR	Positive Jump Relative	6X	XX		S
PRG	Program				P
PTA	Transfer P to A	01	01		N
RAB	Replace Add Backward	53	XX		N
RAC	Replace Add Constant	52	00	XXXX	N
RAD	Replace Add Direct	50	XX		N
RAF	Replace Add Forward	52	XX		N
RAI	Replace Add Indirect	51	XX		N
RAM	Replace Add Memory	51	00	YYYY	N
RAR	Replace Add Relative	5X	XX		S
RAS	Replace Add Specific	53	00		N
REM	Remarks				P
RS1	Right Shift One	01	14		N
RS2	Right Shift Two	01	15		N
SBB	Subtract Backward	37	XX		N
SBC	Subtract Constant	36	00	XXXX	N
SBD	Subtract Direct	34	XX		N
SBF	Subtract Forward	36	XX		N

* Normal codes = N, pseudo-ops = P, and special relative codes = S
**X for NOP cannot be 0

Mnemonic	Name	F	E	G	Type*
SBI	Subtract Indirect	35	XX		N
SBM	Subtract Memory	35	00	YYYY	N
SBN	Subtract No Address	07	XX		N
SBR	Subtract Relative	3X	XX		S
SBS	Subtract Specific	37	00		N
SBUX	Set Buffer Bank Control	01	4X		N
SCB	Selective Complement Backward	17	XX		N
SCC	Selective Complement Constant	16	00	XXXX	N
SCD	Selective Complement Direct	14	XX		N
SCF	Selective Complement Forward	16	XX		N
SCI	Selective Complement Indirect	15	XX		N
SCM	Selective Complement Memory	15	00	YYYY	N
SCN	Selective Complement No Address	03	XX		N
SCR	Selective Complement Relative	1X	XX		S
SCS	Selective Complement Specific	17	00		N
SDCX	Set Direct Bank Control	00	4X		N
SHA	Shift A Left	01			N
SICX	Set Indirect Bank Control	00	2X		N
SIDX	Set Indirect and Direct Bank Control	00	6X		N
SLJX	Selective Jump	77	X0	YYYY	N
SLSX	Selective Stop	77	0X		N
SJS	Selective Stop and Jump	77	XX	YYYY	N
SRB	Shift Replace Backward	47	XX		N
SRC	Shift Replace Constant	46	00	XXXX	N
SRD	Shift Replace Direct	44	XX		N
SRF	Shift Replace Forward	46	XX		N
SRI	Shift Replace Indirect	45	XX		N
SRJX	Set Relative Bank Control and Jump	00	1X		N
SRM	Shift Replace Memory	45	00	YYYY	N
SRR	Shift Replace Relative	4X	XX		S
SRS	Shift Replace Specific	47	00		N
STB	Store Backward	43	XX		N

* Normal codes = N, pseudo-ops = P, and special relative codes = S

Mnemonic	Name	F	E	G	Type*
STC	Store Constant	42	00	XXXX	N
STD	Store Direct	40	XX		N
STEX	BER to Location 6X, A to BER	01	6X		N
STF	Store Forward	42	XX		N
STI	Store Indirect	41	XX		N
STM	Store Memory	41	00	YYYY	N
STPX	P to Location 5X	01	5X		N
STR	Store Relative	4X	XX		S
STS	Store Specific	43	00		N
SUPA	Suppress Listable Output				P
SUPB	Suppress Binary Output				P
WAI	Wait				P
ZJB	Zero Jump Backward	64	XX		N
ZJF	Zero Jump Forward	64	XX		N
ZJR	Zero Jump Relative	6X	XX		S

* Normal codes = N, pseudo-ops = P, and special relative codes = S

APPENDIX B

OPERATING INSTRUCTIONS FOR EXTENDED ASSEMBLY PROCESS (OSAS-AX)

The Extended 160-A Assembly System (OSAS-AX) performs essentially the same function as OSAS-A with the important difference that a much larger symbol table is feasible with the extended version.

Magnetic Tapes

Tape assignments are:	Unit #1 - Symbolic input
	Unit #2 - Intermediate input/output
	Unit #3 - Listable output
	Unit #4 - Binary output

To rewind unload tapes 1, 3 and 4 after final halt, run - DO NOT CLEAR. Tape 1 will not rewind if Selective Jump switch 1 is set.

To Assemble

a) Normal Assembly - no intermediate I/O

1. Master clear
2. Set bank control registers to zero
3. Run
4. At stop 2142, program has been assembled correctly. Run to rewind unload tape units or master clear and run to assemble next program. If stop 2141 occurs, certain lines of symbolic input were in error. The A register displays number of lines in error. Run to stop 2142.

b) Normal Assembly - with intermediate I/O

1. to 3. Same as a)

4. At stop 1142, load intermediate I/O for cards or paper tape (magnetic tape reads back in automatically) by positioning and running.
5. Same as 4. in a)

c) Punch symbol table during assembly

1. Prior to assembling for any option, set Jump switch 4.
2. Proceed with assembly.

d) Halt program after first pass with magnetic tape unit 1 in rewind load position

1. Prior to running assembly program, set Jump switch 1.
2. Proceed with assembly.

3. At stop 1045, first pass is complete and tape unit 1 is in rewind load position. Run from this point unless symbolic input tape is to be saved.

e) Halt assembly program prior to completion

1. Set Stop switch 1. Do NOT take out of run position to stop assembly program.
2. To resume assembly, restore Stop switch 1 and run.
3. To restart assembly program, master clear and run.

NOTE: If OSAS-A is being loaded from a previously prepared bi-octal tape, always load it first with P and A registers at zero before starting assembly procedure.

Stops

Normal Stops:

1045 Will occur only if Jump switch 1 has been set. Occurs after the first pass so that magnetic tapes and tape designations can be changed in a two-tape I/O system.

1142 Position intermediate I/O for input and run.

2142 Final stop. The assembly has been completed correctly. Run if tape units are to be rewound unload. Master clear and run to assemble next program.

2170 Post-final stop occurs after running from stop 2055. Master clear and run to assemble next program.

2267 The WAI pseudo-op has been encountered. Position next symbolic input portion and run to continue assembly. To simulate an END pseudo-op, enter some non-zero quantity into the A register and run.

Error Stops:

2141 Line error display. The number of program lines containing errors is displayed in the A register. Run to proceed to final stop (2055).

0344 Con table - Bank 1 full.

or

4007 Var table - Banks 2 and 3 full.
Symbol table is too large for computer to handle. Either the number of symbols must be decreased, or the program must be assembled in parts; symbols common to more than one part must be equated to each other (using the EQU pseudo-op). Each part (except the first) must be given as origin the address immediately after the last address of the preceding

portion (using the ORG pseudo-op). Maximum symbol table capacity is 1023_{10} constant symbols (Bank 1) and 2046_{10} variable symbols (Banks 2 and 3).

Symbol Table (See page 13)

The symbol table extends over three banks, bank 1 containing 1023_{10} constant symbols and banks 2 and 3 containing $2 \times 1023_{10}$ or 2046_{10} variable symbols.

External symbol tables are dispensed with.

Other publications concerning programming and programming systems for the Control Data Corporation 160-A Computer are:

160-A Reference Manual	#60014500
Satellite Programming	#60018700
160-A FORTRAN/General Information	#60050500
INTERFOR	#60051200
Peripheral Processing Package	#60051700
160-A FORTRAN/Reference Manual	#60051300
AUTOCOMM/Reference Manual	#60051900

CONTROL DATA SALES OFFICES

ALAMOGORDO • ALBUQUERQUE • ATLANTA • BEVERLY HILLS • BOSTON
CAPE KENNEDY • CHICAGO • CLEVELAND • COLORADO SPRINGS • DALLAS • DAYTON
DENVER • DETROIT • DOWNEY • HONOLULU • HOUSTON
HUNTSVILLE • ITHACA • KANSAS CITY, KAN. • MINNEAPOLIS • NEWARK
NEW ORLEANS • NEW YORK CITY • OAKLAND • OMAHA • PHILADELPHIA
PHOENIX • PITTSBURGH • SACRAMENTO • SALT LAKE CITY • SAN BERNARDINO
SAN DIEGO • SAN FRANCISCO • SEATTLE • WASHINGTON, D.C.

INTERNATIONAL OFFICES

FRANKFURT, GERMANY • HAMBURG, GERMANY • STUTTGART, GERMANY
LUCERNE, SWITZERLAND • ZURICH, SWITZERLAND • MELBOURNE, AUSTRALIA
CANBERRA, AUSTRALIA • ATHENS, GREECE • LONDON, ENGLAND • OSLO, NORWAY
PARIS, FRANCE • STOCKHOLM, SWEDEN

CONTROL DATA
CORPORATION

8100 34th AVENUE SOUTH, MINNEAPOLIS 20, MINNESOTA

Pub. No. 60050700
Revision C

Litho in U.S.A.