OPERATING INSTRUCTIONS

The PB-250 Assembler is an IBM-7090 FAP program and is available in column binary deck form which can be stacked and run with other Fortran programs. The PB-250 symbolic program must be punched on cards and treated as a data package. The assembly program uses logical tape units 9 and 10 and scratch tapes should be loaded on these units prior to assembly. A message to the operator to this effect will be printed on the on-line printer at the beginning of the assembly.

INPUT DECK

The PB-250 symbolic program must be arranged as follows:

Card 1 - Fortran Data Card

Asterisk (*) in column 1 and DATA punched in columns 7-10

Card 2 - PB-250 Memory Make-up Card

This card gives the memory capacity of the PB-250 for which this program is being assembled. Columns 1 through N+1 (where N is the highest delay line number present in the PB-250 memory complement) must contain a numeric code designating the lengths of the corresponding memory delay lines as follows:

0	this line not present in this PB-250
1	16 word line
2	32 word line
3	256 word line

The code word in column J describes line number J-1.

Punch a "1" in column 1 and 16, and a "3" in columns 2 through 15 and 17.

Cards 3-N - PB-250 Symbolic Program

Card N+1 - End Card

END punched in columns 8-10. This is the last card of the data package and designates the end of assembly.

ASSEMBLER OUTPUT

The assembler output comes in two forms, a printed list and a deck of punched cards. The printed list contains absolute code, symbolic code, alarms that occurred during assembly and item numbers which are assigned to all cards in the symbolic deck. The punched deck contains the absolute

code in a form suitable for conversion to 5-level paper tape, by means of an IBM 063 or 1401, which can be loaded directly into the PB-250 (see section on Loader). Only those lines which were assembled will be punched on cards in the following format. Punched on the cards will be various combinations of the letters A through R. The letters A through P each represent 4 bit characters 00 through 17 respectively, and Q and R are control characters. An R marks the beginning of a line with the two characters just preceding the R specifying which particular line. A Q signifies the end of pertinent information. The 1536 characters following the R represent the contents of that line. Six characters make up one PB-250 word with the first 6 following the R being the word in sector 000, the second 6 the word in sector 001, etc. The leading two bits of the first character of each group of six have no meaning and are ignored by the loader.

The IBM-063 card to paper tape converter converts each of the 72 columns of each card to its TWX representation and punches this TWX code on 5-level paper tape. The 063 plug board requires only two wires: one from "Card Read On" to "Card Column 1" and one from "Card Column 72" to "Release."

LOADING ASSEMBLED PROGRAM

A PB-250 program is available on 8-level binary tape to load the 5-level program tape into the PB-250. This program is relocatable and can be executed in any command line. The steps to be followed are:

1) Position loader tape in tape reader and type "LLF.". This loads the load routine into line LL.

2) Position 5-level program tape in tape reader and type "000LL.". This will start the load program to executing. Control will be returned to the flexowriter keyboard via the octal utility package at the end of the tape.

The eighteen acceptable characters on the 5-level tape are the TWX representations for the letters A through R, and if illegal characters are encountered during loading, the computer will halt with a line number of 37 indicated on the console lights. Teletype control characters (LTRS, FIGS, CARRIAGE RETURN, and LINE FEED) and blank frames are not considered illegal but are simply ignored.

In case of an illegal character halt, the illegal character can be rejected and loading operation resumed by clearing the parity with the Breakpoint switch. If it is desired to set this character to zero rather than reject it, restart program by typing "112LL.".

ACCEPTABLE OPERATION AND PSEUDO-OPERATION CODES

Normal		Special	Pseudo
CIB	IAC	BSI	END
LAI	LDP	BSO	RSV
RFU	LDC	MAC	DEC
RPT	LDB	IAM	OCT
RTK	LDA	MLX	SET
DIU	TOF	MCL	BRK
HLT	TRU	PTU	
NOP	TCN	TES	
EXF	TBN	SAI	
AWC	TAN	LRS	
AOC	CAM	SBR	
AMC	GTB	SRT	
EBP	CLC	RSI	
STD	CLB	SLT	
STC	CLA	LSD	
STB	DPS	NØR	
STA	DPA	NAD	
ROT	SUB	SQR	
IBC	ADD	DVR	
		DIV	
		MUP	
		WOC	

INTRODUCTION

THE PB-250 SYMBOLIC ASSEMBLER OFFERS THE CONVENIENCE OF PROGRAMMING IN A SYMBOLIC LANGUAGE WHICH VERY CLOSELY RESEMBLES THE PB-250 MACHINE LANGUAGE. THE ASSEMBLER HAS ALL THE FLEXIBILITY OF MACHINE LANGUAGE PROGRAMMING PLUS THE TIME SAVING FEATURE OF PSEUDO-OPERATIONS WHICH WILL GENERATE OTHER LINES OF CODE. THE SYMBOLIC PROGRAM IS RELOCATIBLE AND WILL BE TRANSLATED INTO ABSOLUTE MACHINE CODE BY THE ASSEMBLY PROGRAM.

LANGUAGE

A SYMBOLIC INSTRUCTION WILL ALWAYS CONSIST OF THREE FIELDS, THE TAG FIELD, THE OPERATION FIELD, AND THE FREE FIELD. THE TAG FIELD MAY BE BLANK OR IT MAY CONTAIN A SYMBOLIC TAG. THE OPERATION FIELD WILL CONTAIN THE OP CODE OR THE PSEUDO-OP CODE ALONG WITH THE SEQUENCE AND INDEX TAGS IF THESE ARE USED. ANY OTHER INFORMATION NEEDED TO COMPLETE THE INSTRUCTION MUST BE PLACED IN THE FREE FIELD. THIS WILL INCLUDE SYMBOLIC OR ABSOLUTE ADDRESSES, DECIMAL OR OCTAL INTEGERS, DECIMAL FRACTIONS, SPECIAL SYMBOLS AND COMMENTS.

THE SYMBOLIC INSTRUCTIONS MUST BE PUNCHED ON CARDS IN THE STANDARD SHARE FORMAT. THE TAG FIELD WILL OCCUPY COLUMNS 1 THROUGH 6, THE OPERATION FIELD, COLUMNS 8 THROUGH 14, AND THE FREE FIELD, COLUMNS 16 THROUGH 72. COLUMNS 7 AND 15 MUST BE LEFT BLANK.

COMPATIBLE CHARACTERS

THE PB-250 ASSEMBLER WILL RECOGNIZE THE TEN DECIMAL DIGITS, THE 26 LETTERS OF THE ALPHABET AND THE FOLLOWING SPECIAL CHARACTERS,

.	PERIOD
,	COMMA
+	PLUS
-	MINUS
*	STAR
$	DOLLAR SIGN
=	EQUALS MARK
/	SLASH

TAG FIELD

THE TAG FIELD MAY CONTAIN A SYMBOLIC TAG OR IT MAY BE LEFT BLANK IN WHICH CASE IT IS SAID TO BE UNTAGGED. THE TAG CONSISTS OF UP TO SIX ALPHAMERIC CHARACTERS, THE FIRST OF WHICH MUST BE ONE OF THE LETTERS OF THE ALPHABET. THE REMAINING CHARACTER(S) MAY BE LETTERS OR NUMBERS, FOR EXAMPLE,

N
A31
X3Y2
TAG
TAGA
TAGB
NAME

SPACES IN THE TAG FIELD WILL BE IGNORED. NOTE THAT THE ONLY ALLOWABLE CHARACTERS ARE THE 26 LETTERS OF THE ALPHABET AND THE NUMBERS 0 THROUGH 9, AND IF ANY OTHER CHARACTER OCCURS, THE ALARM NA TAG WILL BE PRINTED IN THE SIDE-BY-SIDE LISTING.

OPERATION FIELD

THE OPERATION FIELD MUST BEGIN IN COLUMN 8 AND IT MAY CONTAIN ANY OF THE NORMAL 3-LETTER PB-250 MNEMONIC OPERATION CODES OR ANY PSEUDO-OPERATIONS ACCEPTABLE TO THE ASSEMBLER. A BLANK OPERATION FIELD WILL CAUSE NO CODE TO BE WRITTEN.

COLUMNS 11 AND 12 OF THE OPERATION FIELD ARE RESERVED FOR THE SEQUENCE AND INDEX TAGS. AN S IN EITHER OF THESE COLUMNS WILL CAUSE THE ASSEMBLER TO PLACE A 1 IN THE SEQUENCE TAG BIT OF THE INSTRUCTION AND AN I IN EITHER COLUMN WILL RESULT IN A 1 PLACED IN THE INDEX TAG BIT. IF EITHER OR BOTH COLUMNS ARE LEFT BLANK, A ZERO WILL BE PLACED IN THE APPROPRIATE BIT(S).

FREE FIELD

THE FREE FIELD MAY BE LEFT BLANK OR MAY CONTAIN ONE OR MORE SUB-FIELDS, DEPENDING ON THE OPERATION. IF MORE THAN ONE SUB-FIELD OCCURS, THE SUB-FIELDS MUST BE SEPARATED BY COMMAS.

EACH SUB-FIELD MAY CONTAIN ONE OR MORE TERMS SEPARATED BY PLUS OR MINUS SIGNS. THE FOLLOWING TERMS MAY BE USED.

1) TAG (CONVERTED TO CORRESPONDING ADDRESS)
2) ABSOLUTE ADDRESS
3) DECIMAL INTEGERS (CONVERTED TO BINARY)
4) *

THE TAG OR * MAY BE FOLLOWED BY A DOLLAR SIGN ($) OR AN EQUALS MARK (=) AN ADDRESS IN LINE ZERO OR LINE 17 RESPECTIVELY

NUMBERS IN THE FREE FIELD WILL BE INTERPRETED AS DECIMAL IF SIGNED AND AS OCTAL IF UNSIGNED.

AN ABSOLUTE ADDRESS WILL CONSIST OF 5 OCTAL DIGITS. THE FIRST 3 DIGITS WILL SPECIFY THE SECTOR NUMBER AND THE LAST 2 WILL DESIGNATE THE LINE NUMBER. SOME EXAMPLES OF LEGAL SUB-FIELDS ARE

00000	LINE 0 SECTOR 0
35301	LINE 1 SECTOR 353
00207	LINE 7 SECTOR 2
00037	LINE 37 SECTOR 0 (INDEX REGISTER)
TAG	THE ABSOLUTE ADDRESS ASSIGNED TO TAG IN COMMAND STORAGE
TAG$	THE SECTOR IN LINE ZERO WHICH CORRESPONDS TO TAG
TAG+1	THE NEXT SEQUENTIAL ADDRESS FOLLOWING TAG IN COMMAND STORAGE
TAG$+9	THIS NINTH SEQUENTIAL ADDRESS FOLLOWING TAG IN THE 16 WORD LINE
*+1	THE NEXT SEQUENTIAL ADDRESS FOLLOWING THE CURRENT INSTRUCTION IN COMMAND STORAGE
*$-1	THE ADDRESS PRECEDING THE CURRENT INSTRUCTION IN THE 16 WORD LINE
TAG-BAG	THE INTEGER THAT RESULTS FROM SUBTRACTING THE ABSOLUTE ADDRESSES ASSIGNED THE TWO TAGS IN COMMAND STORAGE.

ASSIGNING PROGRAM TO MEMORY

THE PB-250 ASSEMBLER ASSIGNS THE PROGRAM TO MEMORY BY SECTIONS RATHER THAN LINE-BY-LINE, AND THE SIZE OF THESE SECTIONS IS DETERMINED BY THE PROGRAMMER. THE PSEUDO-OPERATION BRK NOTIFIES THE ASSEMBLER THAT THIS IS A BREAKPOINT AND THE CURRENT SECTION IS NOW COMPLETE AND CAN BE ASSIGNED TO SEQUENTIAL LOCATIONS IN MEMORY. A TYPICAL SECTION MAY CONTAIN UP TO 256 LINES OF MACHINE CODE, AND AN ALARM WILL BE GIVEN IF THIS LIMIT IS EXCEEDED.

THERE WILL BE OCCASIONS WHEN A PROGRAMMER WOULD LIKE TO SPECIFY THE ABSOLUTE ADDRESS WHERE CERTAIN INSTRUCTIONS ARE TO BE STORED RATHER THAN LEAVE THIS CHOICE TO THE ASSEMBLER. THIS OPTION IS NECESSARY FOR OUTPUT, FOR INSTANCE, WHERE CERTAIN INSTRUCTIONS MUST BE EXECUTED IN LINE 5 FOR FLEXOWRITER OUTPUT AND IN LINE 6 FOR PAPER TAPE OUTPUT. IT WOULD ALSO BE DESIRABLE WHEN WRITING A TIME-OPTIMIZED PROGRAM. THE "SET" PSEUDO-OPERATION IS AVAILABLE TO THE PROGRAMMER FOR THIS PURPOSE. THE SYMBOLIC LINE

```
SET   M
```

INSTRUCTS THE ASSEMBLER TO SET THE LOCATION COUNTER TO THE ABSOLUTE ADDRESS M AND ASSIGN THE NEXT LINE OF CODE TO THAT ADDRESS. ALL SUCCESSIVE LINES OF CODE WILL BE ASSIGNED TO SEQUENTIAL ABSOLUTE LOCATIONS UNTIL A BRK PSEUDO-OPERATION IS ENCOUNTERED.

SECTIONS OF CODE PRECEDED BY A SET PSEUDO-OP WILL BE ASSIGNED TO MEMORY ON THE FIRST PASS OF THE ASSEMBLER AND ALL OTHER CODE WILL BE ASSIGNED ON THE SECOND PASS. DURING THIS OPERATION THE ASSEMBLER KEEPS A RECORD OF AVAILABLE MEMORY IN TABULAR FORM.

TABLE

X1	Y1
X2	Y2
.	.
.	.
.	.
XN	YN

XN WILL BE THE NUMBER OF SEQUENTIAL VACANT LOCATIONS STARTING AT ADDRESS YN, AND THE X,S WILL BE SORTED IN ORDER OF INCREASING MAGNITUDE. AFTER THE ASSEMBLER HAS COUNTED THE NUMBER OF WORDS IN THE CURRENT SECTION, IT WILL SEARCH THE TABLE FOR AN X AT LEAST AS LARGE AS THIS COUNT AND ASSIGN THIS SECTION TO MEMORY STARTING AT THE CORRESPONDING Y. THEN X AND Y WILL BE CORRECTED AND RESTORED TO THE TABLE IN THE PROPER POSITION IF THE NEW X IS GREATER THAN ZERO.

THERE IS ALSO AVAILABLE TO THE PROGRAMMER A PSEUDO-OPERATION WHICH WILL ALLOW HIM TO RESERVE SECTIONS OF MEMORY FOR LATER USE. THE LINE

```
TAG       RSV       N
```

WILL RESERVE N CONSECUTIVE MEMORY LOCATIONS STARTING AT TAG. N MAY BE ANY LEGITIMATE SUB-FIELD, BUT IF TAGS APPEAR IN N THEY MUST HAVE BEEN ASSIGNED PREVIOUS TO THIS POINT IN THE ASSEMBLY.

DATA

MOST PROGRAMS WILL REQUIRE DATA STORED AT DESIGNATED LOCATIONS IN THE MEMORY AND THE ASSEMBLER PROVIDES PSEUDO-OPERATIONS TO PERFORM THIS TASK. THE PB-250 DATA WORD CONSISTS OF A SIGN BIT AND 21 BINARY BITS, WHICH, FOR CONVENIENCE, CAN BE THOUGHT OF AS A SIGN BIT AND 7 OCTAL DIGITS. THE SIGN BIT IS IN THE LEFT-MOST POSITION.

	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1
BINARY	S																					

	7	6	5	4	3	2	1
OCTAL	S						

OCTAL DATA IS USEFUL IN MASKING, COMPARING AND INCREMENTING OPERATIONS AND CAN BE ENTERED BY THE OCT PSEUDO-OP. THE SYMBOLIC LINE

```
TAG     OCT     N
```

WILL TELL THE ASSEMBLER TO CONVERT THE OCTAL INTEGER N TO ITS BINARY EQUIVALENT AND STORE IT IN THE ABSOLUTE ADDRESS IN COMMAND STORAGE ASSIGNED TO TAG. N MAY CONSIST OF UP TO 8 OCTAL DIGITS. IF LESS THAN 8 APPEAR THE OCTAL INTEGER WILL BE RIGHT ADJUSTED AND THE REST OF THE WORD FILLED WITH ZEROS. WHEN 8 DIGITS ARE SPECIFIED IN THE FREE FIELD, THE FIRST ONE MUST BE A ONE OR A ZERO.

SINCE THE PB-250 OPERATES STRICTLY IN FIXED-POINT ARITHMETIC, DECIMAL DATA MUST BE STORED AS SCALED BINARY INTEGERS. AT THIS POINT SOME COMMENTS ON SCALING ARE IN ORDER.

SCALING

THE SCALING OF A DATA WORD MERELY PLACES AN IMAGINARY BINARY POINT AT SOME LOCATION REFERENCED TO A PREVIOUSLY AGREED UPON POSITION WITHIN THE WORD CONFIGURATION, WHICH IN THE PB-250 IS TAKEN TO BE BETWEEN BITS 21 AND 22 OF THE DATA WORD. SO IF A NUMBER IS SCALED ZERO, THE IMAGINARY BINARY POINT IS ASSUMED TO BE BETWEEN THE SIGN BIT AND THE FIRST DATA BIT, AND A POSITIVE SCALING MOVES THIS POINT TO THE RIGHT WHILE A NEGATIVE SCALING MOVES IT TO THE LEFT. A NUMBER WHICH HAS ITS BINARY POINT TO THE RIGHT OF BIT 1 IS SAID TO BE SCALED +21. IT IS IMMEDIATELY EVIDENT THAT ANY NUMBER SCALED LESS THAN OR EQUAL TO ZERO IS LESS THAN 1.

FOR CONVENIENCE LET US ADOPT THE SHORTHAND NOTATION N/S TO MEAN N SCALED S. WE CAN SEE THAT 1.0/0, 2.0/1 OR 25.7/3 ARE IMPOSSIBILITIES. SOME EXAMPLES OF SCALED MACHINE WORDS ARE,

	22	21	20	19	18	17
.5/0	0	1	0	0	0	0

	22	21	20	19	18	17
.5/2	0	0	0	1	0	0

	22	21	20	19	18	17
.5/-1	0	1	0	0	0	0

	22	21	20	19	18	17
1.0/3	0	0	0	1	0	0

	22	21	20	19	18	17
2.0/4	0	0	0	1	0	0

	22	21	20	19	18	17
3.5/3	0	0	1	1	1	0

THE DEC PSEUDO-OPERATION IS PROVIDED TO LOAD DECIMAL DATA DURING ASSEMBLY. THE SYMBOLIC LINE

```
TAG      DEC      +N/+S
```

WILL CONVERT THE DECIMAL NUMBER N TO ITS FIXED POINT BINARY EQUIVALENT SCALED S AND STORE IT IN THE ADDRESS ASSIGNED TO TAG IN COMMAND MEMORY. THE FREE FIELD OF THIS INSTRUCTION MAY CONTAIN MANY SUB-FIELDS EACH OF WHICH MAY CONSIST OF 2 DECIMAL NUMBERS SEPARATED BY A SLASH MARK (/). THE FIRST OF THESE NUMBERS

WILL BE INTERPRETED AS THE ACTUAL NUMERICAL VALUE AND MAY OR MAY NOT INCLUDE A DECIMAL POINT. IF THERE IS NO DECIMAL POINT, THE NUMBERS WILL BE ASSUMED AN INTEGER.

THIS NUMBER WILL BE SCALED THE NUMBER OF PLACES SPECIFIED BY THE DECIMAL INTEGER S WHICH FOLLOWS THE SLASH MARK. IF NO SLASH MARK APPEARS N WILL BE SCALED +21.

BOTH N AND S MUST BE PRECEDED BY A PLUS OR A MINUS SIGN.

THIS CONVENTION HAS BEEN ADOPTED BY THE ASSEMBLER--NUMBERS PRECEDED BY A SIGN WILL BE INTERPRETED AS DECIMAL, AND NUMBERS NOT PRECEDED BY A SIGN ARE ASSUMED TO BE OCTAL.

OPTIMIZATION

THE PB-250 SYMBOLIC ASSEMBLER MAKES NO ATTEMPT TO OPTIMIZE ASSEMBLED PROGRAMS, BUT RATHER LEAVES THIS JOB TO THE PROGRAMMERS.

PROGRAMS TOO LARGE FOR COMMAND LINES

SINCE MACHINE INSTRUCTIONS CAN ONLY BE EXECUTED IN THE 16 COMMAND LINES, PROGRAMS OF CONSIDERABLE LENGTH WILL PROBABLY EXCEED THIS CAPACITY, REQUIRING THAT SECTIONS OF THESE BE STORED IN THE DATA LINES AND TRANSFERRED TO A COMMAND LINE BEFORE EXECUTION. THE PB-250 ASSEMBLER IS CAPABLE OF HANDLING THIS TYPE OF PROGRAM USING LINE 7 AS A BUFFER BETWEEN DATA AND COMMAND LINES.

FIRST, ASSEMBLE THE PROGRAM WITHOUT REGARD TO LENGTH. THE ASSEMBLER WILL PRINT AN ALARM IF LINE 17 IS USED AND ALSO IF LINE 17 IS EXCEEDED. IF LINE 17 IS USED BUT NOT EXCEEDED, THEN THE PROGRAM WILL FIT IN THE COMMAND LINES AND IS ALL ASSEMBLED AND READY TO RUN.

IF, HOWEVER, LINE 17 IS EXCEEDED THEN SOME SECTIONS MUST BE ALLOCATED TO DATA LINE STORAGE WHICH CAN BE ACCOMPLISHED BY THE SDL PSEUDO OPERATION. THE PROGRAMMER CAN PICK CERTAIN SEGMENTS OF HIS PROGRAM, PLACE AN SDL CARD IN FRONT OF EACH OF THEM AND THEN REASSEMBLE. SINCE LINE 7 IS NOW USED AS A BUFFER, IT CANNOT BE USED AS COMMAND LINE STORAGE, SO IF HE STILL GETS THE ALARM THAT LINE 7 IS USED, HE MUST ALLOCATE EVEN MORE SEGMENTS OF HIS PROGRAM TO DATA LINE STORAGE.

HE SHOULD CHOOSE THE SEGMENTS ALLOCATED TO DATA LINE STORAGE WITH CARE BECAUSE THESE WILL REQUIRE CONSIDERABLY MORE TIME FOR EXECTUION. AN EXCELLENT CANDIDATE FOR DATA LINE STORAGE WOULD BE A SEGMENT WHICH WAS EXECUTED ONLY ONCE OR TWICE.

THE SDL PSEUDO-OPERATION WILL INSTRUCT THE ASEMBLER TO STORE THE FOLLOWING SEGMENT, HEADING AND FOOTING IN A DATA LINE, AND THE ASSEMBLER WILL GENERATE AND STORE THESE 3 LINES OF CODE IN THE CURRENT LOCATION IN COMMAND STORAGE,

```
LDCS      *+1
CODEWORD
TRU       32401
```

THE FORMAT OF THE CODE WORD WILL BE,

```
S         L

SECTOR    LINE
FIELD     FIELD
```

WHERE L AND S ARE THE LINE AND SECTOR NUMBERS OF THE DATA LINE WHERE THE FIRST INSTRUCTION OF THE CODE FOLLOWING THE SDL PSEUDO-OP WAS STORED.

SUBROUTINES

IN GENERAL A SUBROUTINE REQUIRES THE RETURN INSTRUCTION STORED IN THE C REGISTER AND THE PARAMETERS STORED IN CONSECUTIVE LOCATIONS BEGINNING AT NAME +2. A TYPICAL FORMAT FOR A SUBROUTINE WILL BE,

```
NAME      STC      NAME+M      STORE RETURN COMMAND
+1        TRU        *+4       JUMP TO START
+2        RSV        3         PARAMETER 1
+3                             PARAMETER 2
+4                             PARAMETER 3
 .         .          .
 .         .          .
 .         .          .
+M        TRU        RETURN    RETURN TO MAIN PROGRAM
          BRK
```

SUBROUTINES WITH ONE OR TWO PARAMETERS WILL ASSUME THESE PARAMETERS TO BE STORED IN THE A AND B REGISTERS IN ORDER TO SAVE RUNNING TIME AND STORAGE LOCATIONS. HOWEVER, REGARDLESS OF THE NUMBER OF PARAMETERS, THE RETURN ADDRESS MUST ALWAYS BE STORED IN THE C REGISTER.

```
          SUB       NAME,A,B,C
```

SUBROUTINE. THIS PSEUDO-OPERATION TELLS THE ASSEMBLER TO CALL IN SUBROUTINE NAME WHICH HAS PARAMETERS A,B, AND C, AND WILL GENERATE THE FOLLOWING LINES OF CODE,

```
          LDA       A
          STA       NAME+2
          LDA       B
          STA       NAME+3
          LDA       C
          STA       NAME+4
          LDCS      *+1
          TRU       *+2
          TRU       NAME
```

GENERALLY A PROGRAMMER CAN SAVE TIME AND SPACE BY LOADING THE SUBROUTINE PARAMETERS (USING THE DOUBLE PRECISION LOAD AND STORE OPERATIONS) BEFORE CALLING THE SUBROUTINE AND THEN TREATING IT AS IF HAD NO PARAMETERS.

VARIABLE LENGTH OPERATIONS

THE LENGTH OF THIS TYPE OF OPERATION, WHICH INCLUDES THE SHIFTING AND SCALING INSTRUCTIONS ALONG WITH THE ARITHMETIC OPERATIONS MUP,DIV,DVR, AND SQR, IS DETERMINED BY THE SECTOR NUMBER OF THE ADDRESS PORTION OF THE WORD IN QUESTION. THE FREE FIELD OF THIS TYPE OF INSTRUCTION SHOULD CONTAIN ONLY ONE SUB-FIELD WHICH WILL BE INTERPRETED BY THE ASSEMBLER AS THE NUMBER OF SECTORS THROUGH WHICH THIS OPERATION WILL BE CONTINUED. THE ASSEMBLER WILL ADD TO THIS NUMBER THE CURRENT VALUE OF THE LOCATION COUNTER AND THEN ADD ONE TO THIS SUM AND PACK THE RESULT IN THE SECTOR NUMBER OF THE INSTRUCTION. SOME EXAMPLES ARE,

```
LSD     +14     LEFT SHIFT AND DECREMENT 14 PLACES
SRT     +10     SHIFT RIGHT 10 PLACES
LRS     1       LOGICAL RIGHT SHIFT 1 PLACE
MUP     +22     FORM FULL PRODUCT IN AB REGISTER
DIV     7       FORM THE 7 BIT QUOTIENT IN B
```

THE FREE FIELD MAY BE LEFT BLANK IN THE MUP,DIV,DVR AND SQR OPERATIONS, IN WHICH CASE THE ASSEMBLER WILL ASSUME A FULL PRODUCT, QUOTIENT OR SQUARE ROOT IS DESIRED. THE ASSEMBLER WILL THEN SUPPLY A +22 FOR THE MUP,DIV, AND DVR AND A +21 FOR THE SQR.

EXCEPTIONS TO NORMAL SYMBOLIC INSTRUCTION

A NORMAL MACHINE INSTRUCTION WILL CONTAIN ONLY ONE SUBFIELD IN THE FREE FIELD WHICH WILL BE CONVERTED TO THE APPROPRIATE SECTOR AND LINE NUMBER, EXCEPT FOR THE VARIABLE LENGTH OPERATIONS WHERE THE LINE NUMBER WILL BE IGNORED. HOWEVER, IN CERTAIN INSTRUCTIONS THE LINE NUMBER HAS A MEANING WHICH IS COMPLETELY REMOVED FROM AN ADDRESS SENSE, AND FOR THESE INSTRUCTIONS IT WILL BE NECESSARY TO SPECIFY TWO SUBFIELDS IN THE FREE FIELD.
THESE EXCEPTIONS ARE LISTED BELOW.

OP FIELD	FREE FIELD
MCL	L,N
MLX	L,N
IAM	L,N
BSI	L,TAG
BSO	L,TAG
PTU	L,TAG
TES	L,TAG
WOC	C,TAG

THE SYMBOLS IN THE FREE FIELD ARE DEFINED AS,

L	A LINE NUMBER
N	THE NUMBER OF SECTORS TO BE MOVED OR INTERCHANGED
C	THE OCTAL CODE REPRESENTATION OF THE CHARACTER TO BE OUTPUT
TAG	ANY LEGAL ADDRESS, SYMBOLIC OR ABSOLUTE. THE SECTOR FIELD OF THE WORD BEING ASSEMBLED AND THE LINE NUMBER WILL BE IGNORED.

```
START LDASI   *+1          PRESET
      STB                   STORE
      STAI    STORE
SET   CLB                  CLEAR UNFINISHED
      LDASI   *+1           WORD AND
      DEC     +2C            SET SHIFT COUNTER
      STDI    WORD            TO 2C
READ  CLAS    *+1          READ
      RPTS    *+4           ONE
      RPT     *+1            CHARACTER
      TES     36,*-1       AND
      TES     36,*-3        LOAD
      CIBS    *-2            INTO
      LAISI   *+1             A
      CCT     37
      CANSI   *+1          REJECT
      CCT     37            LTRS,
      TCFI    READ
      CANSI   *+1
      CCT     33              FIGS,
      TCFI    READ
      CANSI   *+1
      CCT     1C               LINE FEED,
      TCFI    READ
      CANSI   *+1
      CCT     2                 CARRIAGE RETURN
      TCFI    READ
      CANSI   *+1
      CCT                        OR ZERO
      TCFI    READ
      CANSI   *+1          IF A
      CCT     12            DOLLAR SIGN
      TCFI    LINE           TAKE LINE
      CANSI   *+1          IF A W
      CCT     35            TRANSFER CONTROL
      TCF     CCC01          TO KEYBOARD
      CLB                  IF NONE OF THESE
      SLT     +14           SPECIAL CHARACTERS
      ADDSI   *+1            SET UP
      LDDI    TABLE           ADD STORE
      STAI    LDD             LDD COMMAND
      LDPSI   *+1          LOAD UNFINISHED WORD
WORD  RSV     2             AND SHIFT COUNTER
      IAC                  SHIFT TO MAKE ROOM
      LSC     4             FOR THIS CHARACTER
      IAC                    AND DECREMENT COUNTER
LDD   LDDI    TABLE        LOAD CHARACTER INTO C
      TCNI    ERR          TAKE ERR IF ILLEGAL CHARACTER, ELSE
      ADDSI   *+1          MERGE INTO
      CCT     17            UNFINISHED WORD
      TANI    STORE        IF WORD COMPLETE TAKE STORE,ELSE
      STDI    WORD          STORE UNFINISHED WORD AND COUNTER
      TRLI    READ           AND READ NEXT CHARACTER
STORE STB                  STORE FINISHED
      LDAI    *-1           WORD, INCREMENT
      ADDSI   *+1            STB INSTRUCTION
      FLT     1DD             AND
      STAI    STORE            TAKE
      TRLI    SET               SET
LINE  LDAI    WORD         SET UP
```