AN '8008' EDITOR PROGRAM

AUTHOR: ROBERT FINDLEY

I M P O R T A N T N O T I C E

INTRODUCTION

AN EDITOR PROGRAM IS A PROGRAM WHICH ALLOWS THE INPUT, MANIPULATION, AND OUTPUTTING OF TEXT USING A COMPUTER. SUCH A PROGRAM MAY BE USED FOR A VARIETY OF PURPOSES. THE TEXT MIGHT BE PAGES OF A BOOK OR AN ARTICLE THAT AN AUTHOR IS COMPOSING (SUCH AS THE TEXT YOU ARE NOW READING). OR, IT COULD BE THE SOURCE CODE THAT A COMPUTER PROGRAMMER CREATES WHICH MAY EVENTUALLY BE USED AS INPUT TO AN ASSEMBLER PROGRAM (THAT WOULD USE THE SOURCE LISTING TO GENERATE MACHINE CODE). NO MATTER WHAT THE FINAL OUTPUT IS USED FOR, AN "EDITOR" PROGRAM PROVIDES A MEANS FOR USING THE COMPUTER TO ACCEPT, STORE, AND ALLOW ALTERATIONS TO THE TEXT SO THAT LESS TIME AND EFFORT IS SPENT DOING THE MENIAL TASKS SUCH AS ORGANIZING, CORRECTING, AND GENERAL FORMATTING, THAT WOULD OTHERWISE HAVE TO BE DONE BY MANUAL "TYPEWRITER, ERASER AND PAPER" METHODS. WHEN THE TEXT MATERIAL IS IN THE DESIRED FINAL FORMAT, ONE MAY SIMPLY COMMAND THE COMPUTER TO PRINT OR OTHERWISE DISPLAY A FINAL VERSION OF THE TEXT - READY TO BE MAILED AS A LETTER, OR BECOME A PAGE IN A BOOK, OR TO BE FURTHER PROCESSED, SAY BY AN ASSEMBLER PROGRAM. (OR TO BE USED, SAY, AS INPUT TO PROVIDE PRECISE DIRECTIONS TO A "NUMERICALLY CONTROLLED MACHINE!")

AN EDITOR PROGRAM MAY BE THOUGHT OF AS A PROGRAM WHICH ALLOWS THE COMPUTER'S MEMORY TO BE USED AS A COMPOSITION NOTEBOOK INTO WHICH THE WRITER (OR PROGRAMMER) MAY WRITE THEIR TEXT, MAKE ANY DESIRED CHANGES AND CORRECTIONS (AS, FOR INSTANCE, THEIR THOUGHT PROCESSES PROCEED) AND END UP WITH A "PERFECT" COPY (NO "TYPOS," MISSPELLED WORDS, OR OTHER CLERICAL MISTAKES) BEING OUTPUTTED BY THE COMPUTER AT A CONSIDERABLY FASTER RATE THAN MOST TYPIST CAN PERFORM (THOUGH THAT, OF COURSE, WOULD BE A FUNCTION OF THE TYPE OF OUTPUT DEVICE CONNECTED TO THE COMPUTER). THERE IS NO "MAGIC" INVOLVED HERE. THAT IS, IT IS NOT THE COMPUTER THAT IDENTIFIES MISSPELLED WORDS. BUT RATHER, IT IS THE SYSTEM THAT PROVIDES THE TIME SAVING ADVANTAGES. THE COMPUTER GIVES THE WRITER THE CAPABILTY TO QUICKLY ALTER SINGLE LETTERS, WORDS, OR COMPLETE SENTENCES WITHOUT HAVING TO FUSS WITH PENCIL AND ERASER, AND TO REVIEW SECTIONS OF THE "TEXT" AT WILL. THE COMPUTER ALSO PROVIDES A MEANS BY WHICH FINAL COPIES OF THE TEXT MAY RAPIDLY BE PRESENTED. FURTHERMORE, IT IS ALSO POSSIBLE TO HAVE THE "TEXT" IN THE COMPUTER'S MEMORY, TRANSFERRED TO SOME OTHER SORT OF "MASS MEMORY" DEVICE THAT IS MUCH MORE COMPACT THAN, SAY, PAPER, FOR LONG TERM STORAGE OF THE INFORMATION, IN A MANNER THAT WILL ALLOW IT TO BE ACCESSED AT SOME LATER DATE IN THE EVENT "UPDATES" OR ALTERATIONS MIGHT BE DESIRED.

IN ORDER FOR THE READER TO GET A THOROUGH UNDERSTANDING OF HOW AN "EDITOR" PROGRAM CAN BE SO VALUABLE TO ANYONE THAT PREPARES EVEN A MODEST AMOUNT OF TEXTUAL MATERIAL, LET US EXAMINE WHAT IS INVOLVED IN THE TYPICAL PROCESS OF CREATING A "ROUGH DRAFT" OF, SAY, A BOOK OR MAGAZINE ARTICLE, OR PROGRAM LISTING, AND PROCEEDING TO THE FINAL FINISHED COPY. REGARDLESS OF THE PURPOSE WHICH THE TEXT IS TO SERVE, ONE WILL SEE THAT THE STEPS INVOLVED ARE ESSENTIALLY THE SAME.

FIRST, A WRITER SITS DOWN AND PREPARES A "ROUGH DRAFT" TO GET BASIC IDEAS INTO SOME SORT OF CONCRETE FORM, SUCH AS ON A PIECE OF PAPER. (WHY NOT INTO A COMPUTER'S MEMORY BANKS?) NEXT, THE AUTHOR MAY EXPAND UPON CERTAIN AREAS (OR "SCRATCH OUT" OTHERS). THIS PROCESS IS TYPICALLY DONE WITH A TYPEWRITER (DOUBLE SPACED TO ALLOW ROOM FOR CORRECTIONS!) OR PERHAPS IN LONGHAND. EITHER WAY, THE ROUGH DRAFT USUALLY ENDS UP LOOKING AS THOUGH A TWO YEAR OLD GOT HOLD OF IT WITH A PEN IN ONE HAND AND AN ERASER IN THE OTHER! (WHY NOT MAKE SUCH ALTERATIONS WHILE THE TEXT IS RESIDING IN THE COMPUTER'S MEMORY? THE METHOD LEAVES NO "TELL-TALE" MARKS!) FROM THE AUTHOR'S TYPICAL SCRIBBLED ON, CROSSED-OFF, SMUDGE-MARKED MESS OF A ROUGH DRAFT, SOME POOR SOUL (SOMETIMES THE AUTHOR, AND

SOMETIMES A POOR INNOCENT SECRETARY) MUST DECIPHER THE MIX OF ORIGINAL AND MODIFIED MATERIAL AND ATTEMPT TO MAKE A NEAT LOOKING FINAL COPY. SOMETIMES, THE FIRST SUCH ATTEMPT FAILS - A SENTENCE GETS LEFT OUT, A PARAGRAPH IS NOT DELINEATED, AND THE WHOLE PAGE MUST BE TYPED AGAIN!

ONE CAN SEE THAT THIS PROCEDURE MAY REQUIRE THAT THE ENTIRE TEXT BE WRITTEN A NUMBER OF TIMES BEFORE THE FINAL VERSION IS READY. SUCH A METHOD IS TIME CONSUMING, AND THUS EXPENSIVE. IT CAN ALSO BE A RATHER ANNOYING AND FRUSTRATING EXPERIENCE.

THUS, ONE SHOULD HAVE LITTLE DIFFICULTY DISCERNING THAT IF THE INITIAL DRAFT OF THE TEXT IS STORED IN A COMPUTER, AND ONE PROVIDES A MEANS WHEREBY ONE IS ABLE TO MAKE SELECTED CHANGES TO THE TEXT, THUS ELIMINATING THE NEED TO TYPE AN ENTIRE PAGE OR SECTION OF TEXT WHENEVER MINOR CORRECTIONS ARE MADE, THAT ONE CAN CONSIDERABLY REDUCE THE AMOUNT OF WORK INVOLVED IN THE PROCESS. ADDITIONALLY, BY USING A HIGH SPEED OUTPUT DEVICE FROM THE COMPUTER, ONE CAN SIGNIFICANTLY REDUCE THE AMOUNT OF TIME REQUIRED TO PRODUCE A "FINISHED" PAGE. FINALLY, BY USING OTHER TYPES OF COMPUTER I/O DEVICES, SUCH AS A PUNCHED PAPER TAPE MECHANISM, OR MAGNETIC TAPE STORAGE UNIT, ONE MAY SAVE COPIES FOR FUTURE REFERENCE. SUCH COPIES MAY RAPIDLY BE LOADED BACK INTO THE COMPUTER FOR FURTHER MANIPULATIONS AT A LATER DATE IF REQUIRED.

THE ABILITY FOR THE SYSTEM TO TRANSMIT THE TEXT TO A STORAGE DEVICE SUCH AS A MAGNETIC TAPE UNIT, OR A PAPER TAPE PUNCH, MAKES AN EDITOR PROGRAM A VERY EFFECTIVE TOOL FOR PREPARING SOURCE LISTINGS OF PROGRAMS THAT ARE TO BE ASSEMBLED (BY AN "ASSEMBLER" PROGRAM) INTO MACHINE LANGUAGE CODE. WHEN A SOURCE LISTING HAS BEEN PREPARED, IT CAN BE DUMPED TO A STORAGE DEVICE IN A FORM WHICH WILL ENABLE IT TO BE RE-PROCESSED BY AN ASSEMBLER PROGRAM. SHOULD ERRORS IN THE SOURCE PROGRAM BE DETECTED LATER (SUCH AS WHEN THE ASSEMBLER PROGRAM IS PROCESSING THE LISTING) IT BECOMES AN EASY MATTER TO RE-LOAD THE SOURCE LISTING BACK INTO THE COMPUTER TO ALLOW THE OPERATOR TO RE-USE THE EDITOR PROGRAM TO MAKE CORRECTIONS AND PRODUCE A NEW COPY - WITHOUT HAVING TO MANUALLY RE-ENTER THE ENTIRE SOURCE LISTING!

THE BASIC FUNCTIONS AND CAPABILITIES OF AN "EDITOR" PROGRAM

AN EDITOR PROGRAM GENERALLY HAS A VARIETY OF COMMANDS WHICH ENABLES AN OPERATOR TO ENTER TEXT INTO A PORTION OF THE COMPUTER'S MEMORY, TO MAKE CHANGES TO THE TEXT WHEN IT IS IN THE MEMORY, AND TO CONTROL INPUT AND OUTPUT DEVICES SO THAT TEXT MAY BE DISPLAYED, OR RETRIEVED FROM, OR STORED ON EXTERNAL UNITS.

TYPICAL COMMANDS IN AN EDITOR PROGRAM ALLOW THE OPERATOR TO "CLEAR" THE TEXT BUFFER AREA IN MEMORY, TO "APPEND" TEXT TO THE INFORMATION PRESENTLY IN THE COMPUTER'S TEXT AREA, TO INSERT OR DELETE LINES OF TEXT WITHIN THE MEMORY, TO ALTER OR "CHANGE" SPECIFIC LINES WITHIN THE TEXT BUFFER AREA, TO ACCESS AND MODIFY PARTICULAR AREAS WITHIN A "LINE" OF TEXT, TO DISPLAY THE CONTENTS OF THE TEXT BUFFER, AND TO CONTROL I/O DEVICES ASSOCIATED WITH THE SYSTEM.

THE SPECIFIC I/O (INPUT/OUTPUT) DEVICES USED WITH AN EDITOR PROGRAM CAN NATURALLY VARY FROM ONE SYSTEM TO ANOTHER DEPENDING ON WHAT DEVICES A PARTICULAR COMPUTER FACILITY HAS AVAILABLE TO USE. HOWEVER, THE BASIC FUNCTIONS OF AN EDITOR PROGRAM - THAT IS THE INPUTTING, OUTPUTTING, AND MANIPULATION OF TEXT MATERIAL ARE ESSENTIALLY THE SAME. IN THE PROGRAM DESCRIBED IN THIS MANUAL, THE ACTUAL I/O DRIVER ROUTINES FOR INPUTTING AND OUTPUTTING INFORMATION TO THE PROGRAM WILL BE CONSIDERED AS "USER

PROVIDED FUNCTIONS. THE REQUIREMENTS FOR SUCH I/O ROUTINES WILL BE DESCRIBED LATER. THIS APPROACH HAS BEEN TAKEN SO THAT THE EDITOR PACKAGE MAY BE ADAPTED BY INDIVIDUAL READERS TO OPERATE WITH A WIDE VARIETY OF SUCH DEVICES.

THE EDITOR PROGRAM PRESENTED IN THIS MANUAL IS CAPABLE OF PERFORMING THE TYPES OF FUNCTIONS MENTIONED PREVIOUSLY WHILE OPERATING IN AN '8008' BASED MINICOMPUTER SYSTEM WITH A MINIMUM OF 2 K BYTES OF MEMORY. (THE MORE MEMORY AVAILABLE, THE LARGER THE "TEXT BUFFER" AREA!) EACH FUNCTION AND ASSOCIATED ROUTINE(S) IS EXPLAINED IN DETAIL TO ENABLE THE READER TO UNDERSTAND THE OPERATION OF THE PROGRAM. MANY OF THE ROUTINES DEVELOPED FOR THIS TYPE OF PROGRAM MAY HAVE APPLICATIONS IN OTHER TYPES OF PROGRAMS. A DETAILED, HIGHLY COMMENTED LISTING OF EACH SECTION IS PROVIDED AS IT IS PRESENTED. FINALLY, A COMPLETELY ASSEMBLED VERSION OF THE PROGRAM WILL BE PRESENTED IN A MANNER THAT WILL ENABLE A READER TO EASILY ADD CUSTOMIZED I/O CAPABILITY AND IMPLEMENT THE DESCRIBED EDITOR PROGRAM ON AN '8008' BASED SYSTEM. (THOSE THAT MIGHT DESIRE TO IMPLEMENT SUCH A PROGRAM ON OTHER TYPES OF SYSTEMS SHOULD FIND THE INFORMATION PROVIDED HEREIN OF CONSIDERABLE VALUE. FOR INSTANCE, IMPLEMENTING SUCH A PROGRAM ON AN '8080' SYSTEM WOULD REQUIRE LITTLE MORE THAN TRANSLATING THE SOURCE LISTING TO EQUIVALENT '8080' INSTRUCTIONS.)

I/O (INPUT/OUTPUT) CONSIDERATIONS FOR THE EDITOR PROGRAM

BEFORE DISCUSSING THE VARIOUS OPERATING PORTIONS OF THE EDITOR PROGRAM TO BE DESCRIBED, IT IS NECESSARY TO MENTION SEVERAL POINTS ABOUT THE CHARACTER SET USED AND I/O PROGRAMMING CONSIDERATIONS. SINCE THE PRIMARY FUNCTION OF AN EDITOR PROGRAM IS RELATED TO THE PURE HANDLING OF TEXTUAL INFORMATION (ALPHA-NUMERIC), THE FORMAT FOR INPUTTING AND OUTPUTTING INDIVIDUAL CHARACTERS THAT MAKE UP THE TEXT MUST BE WELL UNDERSTOOD.

THE TRANSFER OF CHARACTERS TO AND FROM THE EDITOR PROGRAM DESCRIBED HEREIN IS ASSUMED TO BE IN THE FORM OF "ASCII" ENCODED CHARACTERS. THE "ASCII" CHARACTER SET CONSIST OF A 7 BIT CODE WHICH IS CAPABLE OF DEFINING UP TO 128 "CHARACTERS" OR "FUNCTIONS." IN THE DESCRIBED VERSION OF THE EDITOR PROGRAM, A "SUBSET" CONSISTING OF 74 DIFFERENT CHARACTERS - THE 26 "UPPER CASE" LETTERS OF THE ALPHABET, THE NUMERALS 0 - 9, AND A SOME SYMBOLS AND PUNCTUATION MARKS ARE UTILIZED. OFTEN, WHEN COMMUNICATING WITH AN ASCII ENCODED I/O DEVICE, AN 8'TH BIT IS ADDED TO THE SEVEN BIT ASCII CODE. THIS 8'TH BIT IS OFTEN REFERRED TO AS THE "PARITY" BIT BECAUSE IT CAN BE USED TO SERVE AS AN ERROR DETECTING BIT. MANY I/O DEVICES ARE DESIGNED TO OPERATE WITH EIGHT BITS OF INFORMATION, REGARDLESS OF WHETHER OR NOT "PARITY" ERROR CHECKING METHODS ARE BEING UTILIZED. THE EDITOR PROGRAM DESCRIBED HEREIN ASSUMES THAT THE "ASCII" SUBSET PRESENTED IN THE TABLE ON THE NEXT PAGE IS UTILIZED AND THAT THE "PARITY" POSITION IS ALWAYS IN A LOGIC ONE STATE. READERS WHO DESIRE TO USE I/O DEVICES THAT USE DIFFERENT CODES CAN STILL USE THE EDITOR PROGRAM BY SIMPLY HAVING THEIR I/O ROUTINES PERFORM THE NECESSARY CONVERSIONS BETWEEN THE DEFINED ASCII CODE AND THE CODE UTILIZED BY THEIR EQUIPMENT.

THE EDITOR PROGRAM HAS BEEN CAREFULLY STRUCTURED SO THAT THE ACTUAL I/O ROUTINES ARE SEPARATE FROM THE OPERATING PORTIONS OF THE PROGRAM. THUS THE USER MAY DEVELOP "I/O DRIVER" ROUTINES TO OPERATE WITH A WIDE VARIETY OF DEVICES DEPENDING ON WHAT THE USER HAS AVAILABLE. THE USER MUST SIMPLY INSURE THAT THE SPECIFIC I/O DRIVER ROUTINES UTILIZED MEET THE SPECIFICATIONS THAT WILL BE DESCRIBED BELOW. IF, FOR EXAMPLE, THE KEYBOARD INPUT DEVICE A USER DESIRES TO USE WITH THE PROGRAM, TRANSMITS

CHARACTERS SYMBOLIZED	BINARY CODE	OCTAL REP	CHARACTERS SYMBOLIZED	BINARY CODE	OCTAL REP
A	11 000 001	301	!	10 100 001	241
B	11 000 010	302	"	10 100 010	242
C	11 000 011	303	#	10 100 011	243
D	11 000 100	304	$	10 100 100	244
E	11 000 101	305	%	10 100 101	245
F	11 000 110	306	&	10 100 110	246
G	11 000 111	307	'	10 100 111	247
H	11 001 000	310	(	10 101 000	250
I	11 001 001	311	)	10 101 001	251
J	11 001 010	312	*	10 101 010	252
K	11 001 011	313	+	10 101 011	253
L	11 001 100	314	,	10 101 100	254
M	11 001 101	315	-	10 101 101	255
N	11 001 110	316	.	10 101 110	256
O	11 001 111	317	/	10 101 111	257
P	11 010 000	320	0	10 110 000	260
Q	11 010 001	321	1	10 110 001	261
R	11 010 010	322	2	10 110 010	262
S	11 010 011	323	3	10 110 011	263
T	11 010 100	324	4	10 110 100	264
U	11 010 101	325	5	10 110 101	265
V	11 010 110	326	6	10 110 110	266
W	11 010 111	327	7	10 110 111	267
X	11 011 000	330	8	10 111 000	270
Y	11 011 001	331	9	10 111 001	271
Z	11 011 010	332	:	10 111 010	272
[	11 011 011	333	;	10 111 011	273
\	11 011 100	334	<	10 111 100	274
]	11 011 101	335	=	10 111 101	275
↑	11 011 110	336	>	10 111 110	276
←	11 011 111	337	?	10 111 111	277
SPACE	11 100 000	240	@	11 000 000	300
CTRL D	10 000 100	204	CTRL N	10 001 110	216
CTRL I	10 001 001	211	CTRL S	10 010 011	223
LINE FEED	10 001 010	212	CTRL T	10 010 100	224
CTRL L	10 001 100	214	CTRL U	10 010 101	225
CAR-RET	10 001 101	215	RUB OUT	11 111 111	377

74 CHARACTER ASCII SUBSET

KEYBOARD INPUT DEVICE A USER DESIRES TO USE WITH THE PROGRAM, TRANSMITS BAUDOT CODE RATHER THAN ASCII, THEN THE KEYBOARD INPUT ROUTINE MUST CONVERT THE BAUDOT CODE INTO THE EQUIVALENT ASCII CODE AND RETURN THE ASCII CODE TO THE OPERATING PORTION OF THE EDITOR PROGRAM.

THERE ARE FOUR SEPARATE AND DISTINCT I/O ROUTINES REQUIRED BY THE OPERATING PORTION OF THE EDITOR PROGRAM. THESE ROUTINES SHOULD BE PREPARED AS SUBROUTINES THAT MAY BE CALLED BY THE MAIN PROGRAM. THE ROUTINES REQUIRED WILL SERVE THE FOLLOWING FUNCTIONS. OPERATOR (TYPICALLY A KEYBOARD) INPUT. DISPLAY (TYPICALLY A PRINTER) OUTPUT. BULK STORAGE INPUT (POSSIBLE FROM A WIDE VARIETY OF DEVICES) AND OUTPUT TO BULK STORAGE. THE REQUIREMENTS FOR EACH TYPE OF I/O ROUTINE, AS FAR AS THE MAIN PROGRAM IS CONCERNED, ARE DEFINED NEXT.

OPERATOR INPUT

THE OPERATOR INPUT ROUTINE WHEN CALLED MUST INPUT A SINGLE CHARACTER FROM A DEVICE SUCH AS A KEYBOARD AND RETURN TO THE OPERATING PROGRAM WITH THE ASCII CODE FOR THE INPUTTED CHARACTER IN THE ACCUMULATOR REGISTER OF THE CPU. THE ROUTINE, CREATED BY THE USER, IS FREE TO USE CPU REGISTERS "A" THROUGH "E" FOR IT'S PROCESSING. IF REGISTERS "H" AND "L" MUST BE USED (TO POINT TO A CONVERSION TABLE FOR EXAMPLE) THEIR CONTENTS MUST BE SAVED AND THEN RESTORED TO THEIR ORIGINAL VALUE PRIOR TO RETURNING TO THE CALLING ROUTINE. THE OPERATOR INPUT ROUTINE IS REFERRED TO IN THE EDITOR PROGRAM BY THE LABEL NAME "RCV." THERE ARE THREE POINTS IN THE DESCRIBED EDITOR PROGRAM WHERE "CAL RCV" IS USED TO SIGNIFY A CALL TO THE "OPERATOR INPUT" SUBROUTINE. ONE IS AT THE INSTRUCTION LABELED "IN2" IN THE "INPUT" ROUTINE (TO BE PRESENTED LATER). THE OTHER TWO LOCATIONS WHERE REFERENCES ARE MADE TO THE "RCV" SUBROUTINE ARE IN THE "SEARCH" ROUTINE WHICH WILL BE PRESENTED AS PART OF THE EDITOR PACKAGE.

AN ADDITIONAL FUNCTION THE USER SHOULD PROVIDE IN THE OPERATOR INPUT SUBROUTINE IS THE CAPABILITY TO "ECHO" WHATEVER IS ENTERED BY THE OPERATOR TO A DISPLAY DEVICE. FOR INSTANCE, IF THE OPERATOR INPUT IS COMING FROM THE KEYBOARD OF A ELECTRONIC TYPEWRITER AS KEYS ARE DEPRESSED, IT WILL TYPICALLY BE NECESSARY TO HAVE THE "RCV" ROUTINE SEND BACK THE CODE IT HAS JUST RECEIVED TO THE PRINTER MECHANISM SO THAT THE OPERATOR CAN VERIFY WHAT WAS JUST ENTERED. OR, ONE MIGHT HAVE A SYSTEM WHERE AN ELECTRONIC KEYBOARD WAS BEING USED TO INPUT INFORMATION FROM THE OPERATOR, AND A "TVT" (TELEVISION-TYPE-WRITER) WAS BEING USED TO DISPLAY INFORMATION. IN SUCH A CASE, THE "RCV" SUBROUTINE WOULD NEED TO BE CONFIGURED SO THAT AS INFORMATION WAS SENT TO THE COMPUTER, IT WAS ALSO DISPLAYED ON THE DISPLAY SCREEN.

DISPLAY OUTPUT

THE DISPLAY OUTPUT ROUTINE IS DISTINCT FROM THE "ECHO" ROUTINE DESCRIBED AS PART OF THE OPERATOR INPUT ROUTINE ABOVE. (THOUGH, IN MANY CASES IT MAY BE THAT THE "ECHO" REFERRED TO IS SIMPLY OBTAINED BY THE "RCV" ROUTINE CALLING THE USER'S DISPLAY OUTPUT ROUTINE AS IT IS DEFINED HERE!) THE DISPLAY OUTPUT ROUTINE WHEN CALLED BY THE EDITOR PROGRAM MUST BE CAPABLE OF OUTPUTTING THE ASCII ENCODED CHARACTER CONTAINED IN THE ACCUMUALTOR TO THE DISPLAY DEVICE. THE ROUTINE IS FREE TO USE CPU REGISTERS "B" THROUGH "E" FOR PROCESSING. THE CALLING ROUTINE EXPECTS THE ACCUMULATOR AND REGISTERS "H" AND "L" TO CONTAIN THE ORIGINAL INFORMATION WHEN THE SUBROUTINE IS EXITED. THE DISPLAY OUTPUT SUBROUTINE IS REFERENCED IN THE EDITOR PROGRAM BY A "CAL PRINT" INSTRUCTION. THERE ARE FOUR PLACES WHERE THE "CAL PRINT" COMMAND IS USED. THE "ERROR" SUBROUTINE MAKES REFERENCE TO THE DISPLAY TO PRESENT ERROR MESSAGES. THE SUBROUTINE DESIGNATED "MSG" IN THE PROGRAM USES THE DISPLAY TO PRESENT LISTINGS OF INFORMATION IN THE TEXT BUFFER AND ALSO TO DISPLAY "COMMAND MODES" TO THE OPERATOR. THE "TAB" ROUTINE USES THE DISPLAY OUTPUT SUBROUTINE, AND THE "SEARCH" ROUTINE USES IT TOO.

BULK STORAGE INPUT

THE BULK STORAGE DEVICE INPUT ROUTINE WHEN CALLED MUST INPUT AN ENTIRE "BLOCK" OF TEXT FROM THE BULK STORAGE DEVICE AND STORE THE TEXT IN THE TEXT BUFFER. A "BLOCK" OF TEXT IN THIS INSTANCE IS DEFINED AS THE

ENTIRE CONTENTS OF THE TEXT BUFFER PLUS FOUR "SPECIAL" LOCATIONS THAT ARE LOCATED IN THE FOUR MEMORY BYTES IMMEDIATELY PRECEEDING THE AREA USED BY THE TEXT BUFFER. THESE FOUR "SPECIAL" LOCATIONS ARE USED TO STORE THE LOW AND HIGH ADDRESS OF THE TEXT BUFFER POINTER, AND THE NUMBER OF LINES OF TEXT IN THE TEXT BUFFER. (USE OF THIS INFORMATION WILL BE DESCRIBED LATER AS VARIOUS EDITOR ROUTINES ARE PRESENTED.) REFERENCE TO THE BULK STORAGE INPUT SUBROUTINE IS DENOTED BY THE INSTRUCTION "CAL READ" IN THE EDITOR PROGRAM.

BULK STORAGE OUTPUT

THE BULK STORAGE OUTPUT ROUTINE, WHEN CALLED BY THE INSTRUCTION "CAL PUNCH," MUST OUTPUT THE FOUR BYTES OF MEMORY IMMEDIATELY PRECEEDING THE TEXT BUFFER STORAGE AREA, PLUS THE ENTIRE CONTENTS OF THE TEXT BUFFER, TO THE BULK STORAGE DEVICE AS ONE "BLOCK" OF DATA. THE ROUTINE IS FREE TO USE ALL OF THE CPU REGISTERS FOR IT'S OPERATION WITH THE CONSIDERATION THAT WHEN THE ROUTINE IS INITIALLY CALLED, CPU REGISTERS "H" AND "L" WILL BE POINTING TO THE FIRST BYTE OF MEMORY IN THE BLOCK AND CPU REGISTERS "D" AND "E" WILL BE POINTING TO THE LAST BYTE OF THE TEXT BUFFER THAT IS TO BE TRANSMITTED.

I/O INTEGRITY CONSIDERATIONS

THE OPTION OF PERFORMING ERROR CHECKS ON THE TRANSMISSION OF DATA TO AND FROM THE PERIPHERAL DEVICES IS LEFT TO THE USER. THIS IS DONE BECAUSE THERE ARE A VARIETY OF ERROR CHECKING TECHNIQUES POSSIBLE, DEPENDING ON THE TYPE OF DEVICE BEING USED IN THE SYSTEM. FOR EXAMPLE, A USER WITH A PAPER TAPE READER SYSTEM MAY ELECT TO PROVIDE FOR PARITY CHECKING TECHNIQUES. SUCH TECHNIQUES MAY BE IMPLEMENTED USING "EVEN" OR "ODD" PARITY CONVENTIONS DEPENDING ON THE TYPE OF DEVICE, OR EVEN THE USER'S PREFERENCE. ANOTHER TYPE IF I/O DEVICE, SUCH AS A COMMERCIAL MAGNETIC TAPE, OR DISC UNIT, MAY HAVE AUTOMATIC "BLOCK" ERROR CHECKING CAPABILITIES, IN WHICH CASE THE USER WOULD WANT TO HAVE THE APPROPRIATE I/O ROUTINE TEST FOR ERROR CONDITIONS AND TAKE APPROPRIATE ACTION. THE USER MAY ELECT, IF ERROR CHECKING CAPABILITIES ARE IMPLEMENTED, TO ADD ADDITIONAL ROUTINES THAT PRESENT ERROR MESSAGES TO THE OPERATOR, OR THAT DIRECT THE OPERATION OF "ERROR CORRECTING" TECHNIQUES. IN ANY EVENT, SUCH TECHNIQUES ARE OUTSIDE THE SCOPE OF THIS PARTICULAR PUBLICATION AND WILL BE LEFT TO THE USER TO IMPLEMENT AS DESIRED.

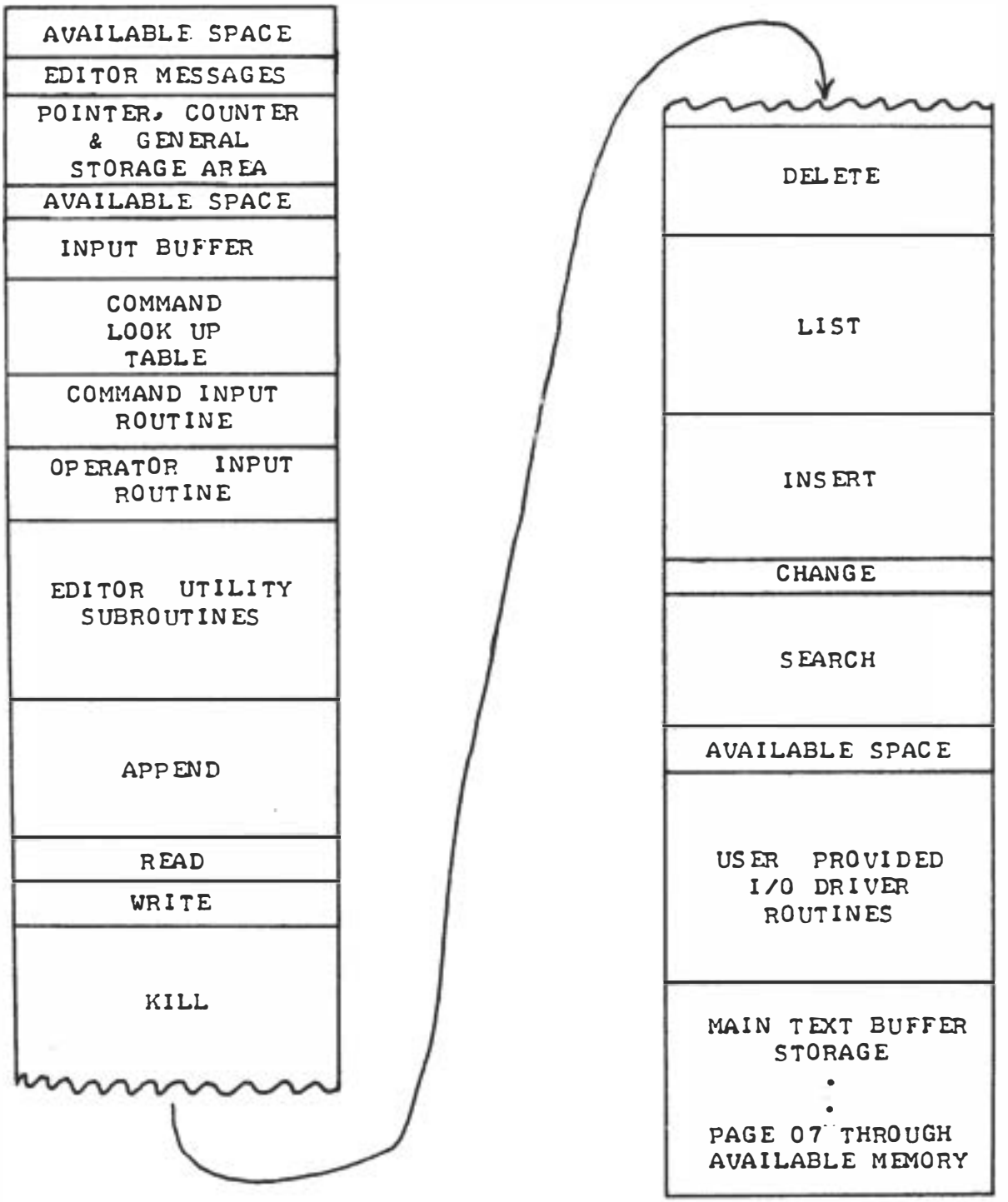
MEMORY UTILIZATION OF THE EDITOR PROGRAM

THE EDITOR PROGRAM PRESENTED IN THIS MANUAL OPTIMIZES THE UTILIZATION OF MEMORY FOR BOTH THE STORAGE OF TEXT AND THE STORAGE OF THE OPERATING PROGRAM ITSELF. THE MANNER OF DEFINING AND STORING LINES OF TEXT IN THE TEXT BUFFER ALLOWS THE MAXIMUM USAGE OF THE TEXT BUFFER AREA AND MINIMIZES THE AMOUNT OF SPACE REQUIRED BY THE OPERATING PROGRAM. MEMORY USEAGE IS AS FOLLOWS.

THE OPERATING PROGRAM PROPER RESIDES ON PAGES 01 THROUGH 05 OF MEMORY. ADDITIONALLY, THE OPERATING PROGRAM ASSUMES THAT USER PROVIDED I/O ROUTINES WILL RESIDE ON PAGE 06. PORTIONS OF PAGE 00 ARE USED AS A "SCRATCH PAD" AREA FOR THE PROGRAM IN WHICH POINTERS, COUNTERS, AND TEMPORARY DATA IS MAINTAINED. ANOTHER PORTION OF PAGE 00 CONTAINS SEVERAL "MESSAGE STRINGS" USED BY THE PROGRAM. THE LATTER HALF OF PAGE 00 IS USED AS A TEMPORARY TEXT INPUT BUFFER. THE ACTUAL LENGTH OF THE TEXT

INPUT BUFFER MAY BE VARIED FROM 1 TO 128 (DECIMAL) CHARACTERS TO ACCOMMODATE VARIOUS TYPES OF I/O DEVICES. THE MEANS OF VARYING THE INPUT BUFFER LENGTH WILL BE DESCRIBED LATER.

THE FINAL MAJOR BLOCK OF MEMORY IS THAT AREA USED TO ACTUALLY STORE THE TEXT - THE TEXT BUFFER AREA. THE TEXT BUFFER AREA STARTS ON PAGE 07 AND EXTENDS THROUGH THE LAST AVAILABLE PAGE OF RAM (READ AND WRITE MEMORY) IN THE SYSTEM. A MEMORY MAP ILLUSTRATING MEMORY USEAGE FOR THE DESCRIBED PROGRAM IS ILLUSTRATED BELOW. IT MAY BE NOTED FROM THE MEMORY MAP THAT THERE ARE A FEW UNUSED MEMORY LOCATIONS IN THE PROGRAM. THESE SPACES MAY BE USED FOR I/O DRIVER ROUTINES SHOULD THE USER FIND THAT MORE THAN ONE PAGE (ALLOCATED ON PAGE 06) IS REQUIRED, OR IF ADDITIONAL FUNCTIONS ARE ADDED TO THE FUNDAMENTAL EDITOR PROGRAM. SOME COMMENTS ON SUCH OPTIONS WILL BE PROVIDED LATER.



FUNDAMENTAL OPERATION OF THE EDITOR PROGRAM

THE EDITOR PROGRAM IS CONCEPTUALLY QUITE SIMPLE IN OPERATION. ESSENTIALLY, THE PROGRAM CONSIST OF A SERIES OF ROUTINES FOR CONTROLING TWO "TEXT BUFFER" AREAS. ONE SUCH AREA, ON THE LATTER HALF OF PAGE 00, IS USED TO STORE A "OPERATOR COMMAND" AND A SINGLE LINE OF TEXT BEING INPUTTED FROM THE "OPERATOR INPUT DEVICE." THE OTHER AREA IS THE LARGE "TEXT BUFFER" WHICH HOLDS TEXT STRINGS AFTER THEY HAVE BEEN ACCEPTED BY THE "TEXT INPUT" BUFFER.

THE TEXT INPUT BUFFER STORES ALL THE CHARACTERS ON A SINGLE LINE AS THEY ARE RECEIVED FROM THE OPERATOR'S INPUT DEVICE. A "LINE" IS NORMALLY DEFINED AS BEING EQUAL TO THE NUMBER OF CHARACTERS THAT CAN BE DISPLAYED ON THE OUTPUT DISPLAY DEVICE BEING USED WITH THE SYSTEM. AS EACH LINE IS TERMINATED (BY A TERMINATION CHARACTER - THE "CARRIAGE-RETURN"), THE STARTING AND ENDING ADDRESS OF THE TEXT STRING IN THE TEXT INPUT BUFFER IS SAVED. THE VALUES SAVED ARE LATER USED BY VARIOUS "COMMAND" ROUTINES TO DETERMINE THE ACTUAL LENGTH (NUMBER OF CHARACTERS) OF A LINE RESIDING IN THE TEXT INPUT BUFFER.

THE EDITOR PROGRAM THEN TRANSFERS INFORMATION IN THE TEXT INPUT BUFFER TO THE LARGER "TEXT BUFFER" IN ACCORDANCE WITH DIRECTIONS IT RECEIVES IN THE FORM OF OPERATOR COMMANDS. LINES OF INFORMATION STORED IN THE LARGE TEXT BUFFER CAN ALSO BE MANIPULATED BY VARIOUS TYPES OF COMMANDS, TO, FOR INSTANCE, INSERT, OR DELETE LINES. THE TEXT BUFFER HAS BEEN LOCATED ABOVE THE OPERATING PORTION OF THE PROGRAM SO THAT THE TEXT BUFFER AREA CAN BE EXPANDED READILY AS A USER ACQUIRES MORE MEMORY FOR A SYSTEM. AS EACH LINE IS TRANSFERRED FROM THE TEXT INPUT BUFFER TO THE MAIN TEXT BUFFER AREA, A "ZERO BYTE" IS ADDED IMMEDIATELY FOLLOWING THE LAST CHARACTER IN THE STRING TO SERVE AS AN "END OF LINE TERMINATOR." THIS ZERO BYTE ALLOWS THE END OF ANY LINE TO BE READILY DETERMINED BY THE OPERATING PORTION OF THE PROGRAM AND DOES NOT CONFLICT WITH STORING OF TEXT AS NONE OF THE DEFINED ALLOWABLE CODES THAT REPRESENT TEXT CHARACTERS HAS AN EIGHT BIT CODE OF ALL ZEROS.

ASSOCIATED WITH THE MAIN TEXT BUFFER IS A TWO REGISTER "POINTER" WHICH IS CONVENIENTLY REFERRED TO HEREIN AS THE "TEXT BUFFER POINTER." THIS POINTER INDICATES THE ADDRESS OF THE LOCATION IMMEDIATELY FOLLOWING THE LAST LINE TERMINATOR IN THE MAIN TEXT BUFFER AREA - OR, TO PUT IT ANOTHER WAY, THE ADDRESS WHERE THE NEXT LINE OF INFORMATION TO BE ADDED TO THE MAIN TEXT BUFFER SHOULD BE PLACED.

ALSO ASSOCIATED WITH THE MAIN TEXT BUFFER IS A TWO BYTE COUNTER. THIS COUNTER WILL BE REFERRED TO AS THE "TEXT BUFFER LINE COUNTER." AS MIGHT BE SURMISED BY IT'S NOMENCLATURE, THE "TEXT BUFFER LINE COUNTER" IS USED TO MAINTAIN A COUNT OF THE NUMBER OF LINES CURRENTLY IN THE TEXT BUFFER!

THE TWO SETS OF REGISTERS, THE "TEXT BUFFER POINTER" PAIR, AND THE "TEXT BUFFER LINE COUNTER" PAIR, ARE THE PRINCIPAL TEXT BUFFER "STATUS" INDICATORS. AS LINES ARE ADDED OR DELETED, OR MODIFIED; THE VALUES OF THESE TWO PAIRS OF REGISTERS ARE CHANGED ACCORDINGLY. THE READER WILL SOON LEARN MORE ABOUT THE DETAILED OPERATIONS OF THE FUNDAMENTAL PORTIONS OF THE EDITOR PROGRAM.

EDITOR COMMANDS

THE OVER-ALL OPERATION OF THE EDITOR PROGRAM IS CONTROLLED BY THE

OPERATOR ENTERING "COMMANDS" ON THE "OPERATOR INPUT DEVICE." THESE COMMANDS DIRECT THE EDITOR PROGRAM TO PERFORM SPECIFIED OPERATIONS VIA APPROPRIATE ROUTINES. THE FORMAT IN WHICH COMMANDS ARE ENTERED MAY VARY DEPENDING ON WHETHER ONE OR MORE LINES OF INFORMATION IN THE TEXT BUFFER WILL BE AFFECTED BY THE COMMAND. THE FOLLOWING IS A SUMMARY OF THE DIFFERENT TYPES OF COMMANDS USED BY THE EDITOR PROGRAM ALONG WITH A BRIEF DESCRIPTION OF THE ASSOCIATED OPERATIONS.

"APPEND" (A) - APPENDS THE TEXT ENTERED BY THE OPERATOR TO THE TEXT BUFFER. TEXT ENTERED IS ADDED TO THE BUFFER IMMEDIATELY FOLLOWING THE LAST "LINE" STORED IN THE BUFFER AT THE TIME THE COMMAND IS ISSUED.

"CHANGE" (C) - CHANGES THE LINE OR LINES SPECIFIED IN THE COMMAND BY DELETING THE SPECIFIED LINE(S) AND REPLACING THE LINE(S) WITH THE INFORMATION ENTERED BY THE OPERATOR FOLLOWING THE ISSUANCE OF THE COMMAND.

"DELETE" (D) - DELETES THE LINE(S) SPECIFIED IN THE COMMAND FROM THE TEXT BUFFER.

"INSERT" (I) - INSERTS LINE(S) OF TEXT IMMEDIATELY PRIOR TO THE LINE OF TEXT SPECIFIED IN THE COMMAND INTO THE TEXT BUFFER.

"KILL" (K) - EFFECTIVELY "CLEARS" THE TEXT BUFFER AREA BY RESETTING THE EDITOR "TEXT BUFFER POINTER" AND THE "TEXT BUFFER LINE COUNTER" IN PREPARATION FOR BEGINNING THE PROCESS OF STORING LINES OF TEXT IN THE TEXT BUFFER.

"LIST" (L) - CAUSES THE LINE(S) SPECIFIED IN THE COMMAND TO BE DISPLAYED ON AN OUTPUT DEVICE.

"SEARCH" (S) - CAUSES THE EDITOR PROGRAM TO SEARCH THE LINE SPECIFIED IN THE COMMAND FOR A SPECIFIC CHARACTER CHOSEN BY THE OPERATOR. THE CONTENTS OF THE LINE BEING SEARCHED IS DISPLAYED UP TO THE POINT WHERE THE CHARACTER BEING SEARCHED FOR IS LOCATED. AT THAT TIME, THE REMAINING CONTENTS OF THE LINE MAY BE REVISED. THE REVISED LINE BECOMES THE NEW LINE IN PLACE OF THE ORIGINAL IN THE TEXT BUFFER.

"READ" (R) - CALLS UPON THE BULK STORAGE INPUT DEVICE TO INPUT TEXT FROM THE DEVICE DIRECTLY INTO THE MAIN TEXT BUFFER.

"WRITE" (W) - CALLS UPON THE BULK STORAGE OUTPUT DEVICE TO STORE THE ENTIRE CONTENTS OF THE TEXT BUFFER ON THE BULK STORAGE OUTPUT DEVICE.

EACH OF THE ABOVE COMMANDS IS ENTERED BY THE OPERATOR ENTERING THE FIRST CHARACTER OF THE COMMAND NAME (ILLUSTRATED IN PARENTHESIS ABOVE). SOME COMMANDS EITHER REQUIRE ("CHANGE," "DELETE," "INSERT" AND "SEARCH") OR PROVIDE THE OPTION ("LIST") OF SPECIFYING THE LINE NUMBER, OR A GROUP OF LINE NUMBERS THAT ARE TO BE AFFECTED BY THE COMMAND. THERE ARE THREE LINE NUMBER SPECIFYING FORMATS THAT MAY BE USED WITH VARIOUS COMMANDS.

LINE NUMBERS FORMAT	ASSOCIATED COMMANDS
Z	(A,K,L,R,W)
Z XXX YYY	(C,D,I,L,S)
Z XXX YYY,MMM NNN	(C,D,L)

WHERE "Z" IS THE COMMAND ABBREVIATION, "XXX YYY" IS A SPECIFIC LINE NUMBER, AND "XXX YYY,MMM NNN" IS A GROUP OF LINE NUMBERS. THE ABBREVIATIONS GIVEN UNDER THE "ASSOCIATED COMMANDS" HEADING INDICATE THE TYPES OF COMMANDS THAT MAY USE EACH FORMAT.

THE FORMAT ILLUSTRATED FOR THE LINE NUMBERS INDICATES THAT EACH LINE IN THE TEXT BUFFER IS REPRESENTED BY TWO GROUPS OF THREE DIGITS. THIS IS BECAUSE THE EDITOR PROGRAM HAS BEEN DEVELOPED TO USE OCTAL NUMBER NOTATION. (THIS METHOD OF LINE NUMBER NOTATION CONSERVES CONSIDERABLE SPACE AS IT ELIMINATES THE NEED FOR DECIMAL TO BINARY AND BINARY TO DECIMAL CONVERSION ROUTINES IN THE EDITOR PROGRAM.) EACH THREE DIGIT GROUP MAY HAVE A VALUE FROM 000 TO 377, WHERE EACH DIGIT IS AN OCTAL VALUE. BY USING TWO THREE DIGIT GROUPS, LINE NUMBERS FROM "000 001" TO "377 377" MAY BE SPECIFIED. THIS RANGE IS EQUIVALENT TO A DECIMAL RANGE OF 65,535 LINE NUMBERS. AT FIRST GLANCE, THE NUMBERING SYSTEM MAY SEEM A BIT AWKWARD. BUT, WITH A LITTLE USE, THE AVERAGE MICROCOMPUTER USER WILL FIND THE SYSTEM QUITE COMFORTABLE, PARTICULARLY SINCE IT IS IN LINE WITH THE POPULAR NUMBERING CONVENTION USED FOR DESIGNATING MEMORY ADRESSES WITH WHICH MOST SMALL COMPUTER USERS ARE FAMILIAR. OF COURSE, IF THE AMBITIOUS READER CAN NOT STAND TO WORK WITH SUCH NOTATION, ONE COULD CONSIDER MODIFYING THE PROGRAM (IT IS NOT TOO DIFFICULT) TO PERFORM THE NECESSARY CONVERSIONS.

IN THE EVENT SOME READERS ARE IN DOUBT ABOUT THE LINE NUMBERING CONVENTION USED IN THE EDITOR PROGRAM BEING DESCRIBED, THE FOLLOWING EXAMPLE ILLUSTRATES HOW LINES ARE NUMBERED IN OCTAL FORMAT AS COMPARED TO DECIMAL NOTATION.

DECIMAL COUNT	OCTAL COUNT	HYPOTHETICAL LINE CONTENTS
1	000 001	THIS IS A SAMPLE
2	000 002	USED TO ILLUSTRATE
3	000 003	THE USE OF THE
4	000 004	OCTAL LINE NUMBERING
5	000 005	SYSTEM AS UTILIZED
6	000 006	IN THE EDITOR PROGRAM
7	000 007	BEING DESCRIBED
8	000 010	NOTE THAT THERE
9	000 011	IS NO LINE NUMBER
10	000 012	8 OR 9 DEFINED IN THE
11	000 013	OCTAL NUMBERING SYSTEM!

A USER UNINITIATED TO THE USE OF THE OCTAL NUMBERING SYSTEM MUST SIMPLY LEARN TO COUNT LINES BY SKIPPING THE NUMBERS "8" AND "9." ELSE ONE MAY BE SURPRISED TO SEE THAT ISSUEING THE COMMAND:

L 000 007,000 010

DOES NOT RESULT IN 4 LINES BEING DISPLAYED - IT RESULTS IN JUST TWO!

GENERAL FLOW OF OPERATIONS FOR THE EDITOR PROGRAM

A GENERAL FLOW CHART OF THE EDITOR PROGRAM DESCRIBED IN THIS MANUAL IS PRESENTED ON THE FOLLOWING PAGE TO PROVIDE AN OVER-VIEW OF THE PROGRAM'S BASIC OPERATION. THE COMMANDS WHICH ARE SHOWN BRANCHING TO THE LEFT SIMPLY PERFORM THE SPECIFIED FUNCTION AND THEN RETURN TO THE EDITOR "COMMAND" MODE. THE COMMANDS WHICH ARE SHOWN BRANCHING TO THE RIGHT IN THE DIAGRAM, CALL THE "OPERATOR INPUT" SUBROUTINE ONE OR MORE TIMES TO INPUT NEW TEXT, OR TO PROVIDE REVISIONS. THE READER SHOULD REFER TO THE GENERAL FLOW CHART FROM TIME-TO-TIME AS THE DIFFERENT ROUTINES IN THE EDITOR PROGRAM ARE PRESENTED TO SEE WHERE THEY FIT IN THE OVER-ALL VIEW.

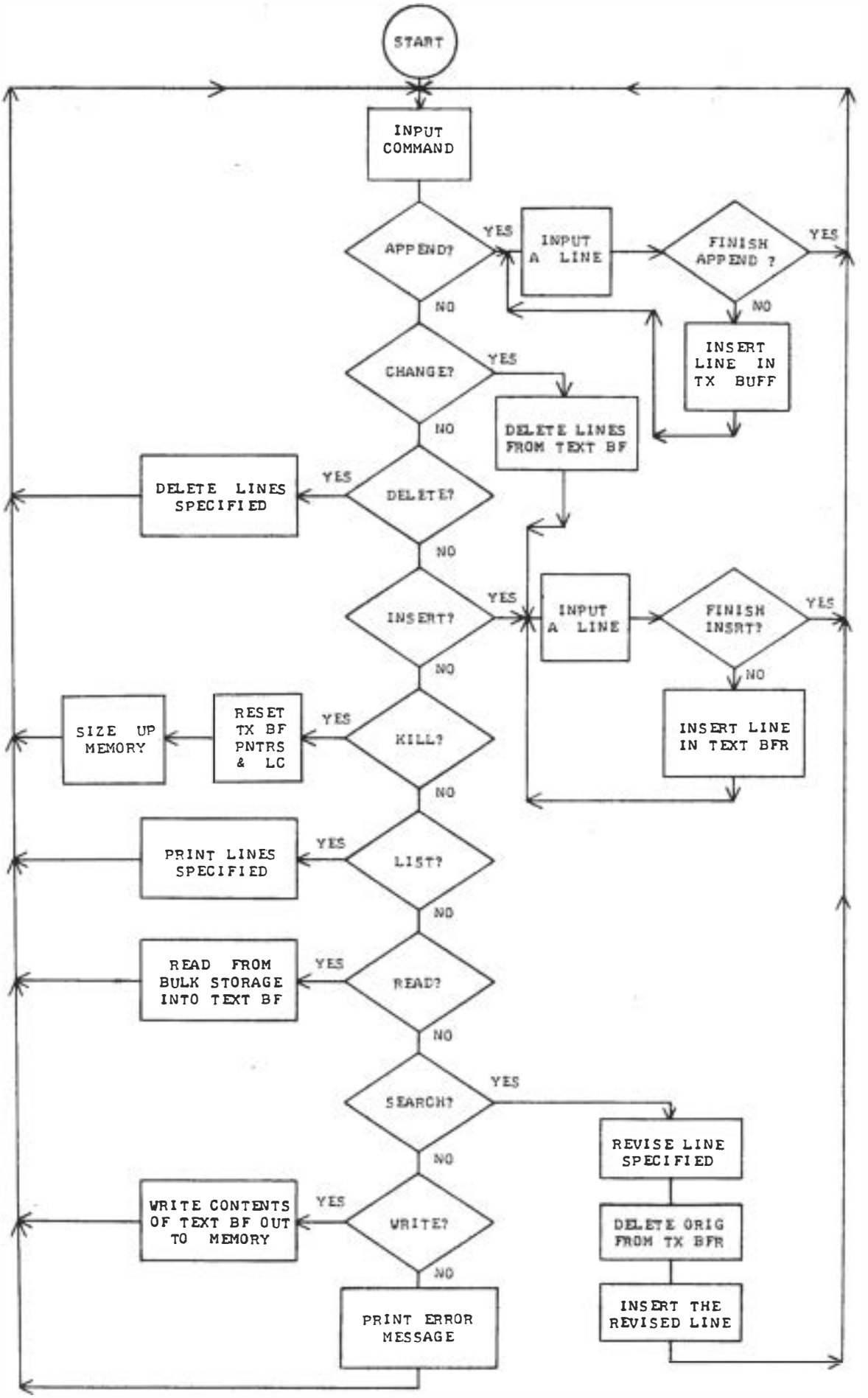
GENERAL UTILITY SUBROUTINES

THERE ARE A GROUP OF SUBROUTINES USED BY MANY OF THE MAJOR ROUTINES IN THE EDITOR PROGRAM. SUCH SMALL SEQUENCES OF INSTRUCTIONS ARE REFERRED TO AS "UTILITY" ROUTINES IN THIS PUBLICATION BECAUSE OF THEIR BROAD, GENERAL USEAGE THROUGH-OUT THE PROGRAM. THESE SUBROUTINES WILL BE PRESENTED HERE TO POINT OUT IMPORTANT ASPECTS RELATING TO THEM AND TO PROVIDE A BASE FROM WHICH TO PRESENT THE MAJOR PORTIONS OF THE PROGRAM. WHILE THE "UTILITY" ROUTINES PRESENTED HERE WERE DESIGNED SPECIFICALLY FOR USE IN THE EDITOR PROGRAM BEING DISCUSSED, MANY READERS MAY FIND THAT SOME OF THE SUBROUTINES MIGHT HAVE APPLICATIONS IN OTHER TYPES OF PROGRAMS.

THE FIRST SUCH "UTILITY" ROUTINE TO BE ILLUSTRATED HAS BEEN DESIGNATED BY THE LABEL "FBFLM" (FETCH BUFFER LIMIT!). THE FUNCTION OF THIS SUBROUTINE IS TO DEFINE THE START ADDRESS FOR THE "TEXT INPUT BUFFER." THE SUBROUTINE APPEARS AS FOLLOWS.

```
   MNEMONIC                COMMENTS
-------------           --------------

FBFLM, LEL              /SAVE INP PNTR
LBI 270                 /LOAD INP BFR LIMIT
LHI 000                 /RESTORE INP PNTR
RET                     /RET TO CALLING PGM
```

AS MENTIONED EARLIER, THE LENGTH OF THE TEXT INPUT BUFFER MAY BE SET (BY THE READER MODIFYING THE ABOVE SUBROUTINE) TO BE FROM 1 TO 128 (DECIMAL) CHARACTERS (BYTES OF MEMORY!) IN LENGTH. THE LENGTH OF THE TEXT INPUT BUFFER SHOULD GENERALLY BE SET TO THE MAXIMUM NUMBER OF CHARACTERS THAT CAN BE DISPLAYED ON THE OUTPUT DEVICE BEING UTILIZED WITH THE EDITOR SYSTEM. DIFFERENT OUTPUT DEVICES TYPICALLY HAVE DIFFERENT MAXIMUMS. FOR INSTANCE, ONE MODEL OF AN ELECTRO-MECHANICAL TYPEWRITER IS TYPICALLY CAPABLE OF TYPING A MAXIMUM OF 72 (DECIMAL) CHARACTERS ON A LINE. HOWEVER, "TVT" (TELEVISION TYPE WRITERS) DISPLAY SYSTEMS GENERALLY CAN ONLY DISPLAY 32 (OR SOMETIMES 48 OR 64) CHARACTERS ON A LINE. THE VALUE OF "270" IN THE "LBI" INSTRUCTION IN THE ABOVE SUBROUTINE SETS A MAXIMUM INPUT TEXT BUFFER LENGTH OF 72 (DECIMAL) OR 110 (OCTAL) BYTES. REMEMBER, THE VALUE "270" IS BEING USED TO DEFINE THE "STARTING" ADDRESS OF THE INPUT TEXT BUFFER. THE INPUT TEXT BUFFER WILL ALWAYS BE CONSIDERED TO EXTEND FROM THE STARTING LOCATION DEFINED BY THE ABOVE SUBROUTINE (ON PAGE 00) ON UP TO LOCATION 377 ON PAGE 00. IF, FOR INSTANCE,


START
INPUT COMMAND
APPEND?
YES
INPUT A LINE
FINISH APPEND ?
YES
NO
INSERT LINE IN TX BUFF
NO
CHANGE?
YES
DELETE LINES FROM TEXT BF
NO
DELETE?
YES
DELETE LINES SPECIFIED
NO
INSERT?
YES
INPUT A LINE
FINISH INSRT?
YES
NO
INSERT LINE IN TEXT BFR
NO
KILL?
YES
RESET TX BF PNTRS & LC
SIZE UP MEMORY
NO
LIST?
YES
PRINT LINES SPECIFIED
NO
READ?
YES
READ FROM BULK STORAGE INTO TEXT BF
NO
SEARCH?
YES
REVISE LINE SPECIFIED
DELETE ORIG FROM TX BFR
INSERT THE REVISED LINE
NO
WRITE?
YES
WRITE CONTENTS OF TEXT BF OUT TO MEMORY
NO
PRINT ERROR MESSAGE

THE USER WANTED TO USE THE EDITOR WITH AN OUTPUT DEVICE THAT DISPLAYED A MAXIMUM OF 32 (DECIMAL) CHARACTERS (40 OCTAL) THEN THE VALUE IN THE "LBI" INSTRUCTION WOULD BE SET TO 340.

THE "LEL" AND "LHI 000" INSTRUCTIONS IN THE "FBFLM" SUBROUTINE ARE FREQUENTLY REQUIRED BY OTHER ROUTINES WHEN THEY OBTAIN THE BUFFER LENGTH VALUE. TO SAVE SPACE IN THE PROGRAM, THEY ARE INCLUDED IN THE SUBROUTINE INSTEAD OF HAVING THEM REPEATED NUMEROUS TIMES THROUGH-OUT THE OTHER PORTIONS OF THE PROGRAM.

THE FOLLOWING GROUP OF UTILITY SUBROUTINES PERFORM SUCH GENERAL CHORES AS INCREMENTING REGISTER PAIRS, SHIFTING THE CONTENTS OF A BYTE OF MEMORY AT ONE ADDRESS TO ANOTHER ADDRESS, INCREMENTING OR DECREMENTING A DOUBLE PRECISION VALUE STORED IN MEMORY, AND SAVING CPU REGISTERS "B" THROUGH "E" IN A SECTION OF MEMORY. THESE TYPES OF ROUTINES MAY BE "OLD HAT" TO MANY READERS, BUT SOME NOVICES MIGHT FIND A FEW NEW TECHNIQUES TO ADD TO THEIR PROGRAMMING REPERTOIRE FOR OTHER APPLICATIONS.

```
    MNEMONIC                COMMENTS
 --------------          --------------

INMEM, INL              /INCR LO ADDR
RFZ                     /RET IF NOT ZERO
INH                     /INCR PAGE REG
RET                     /RET TO CALLING PGM
/
SHFT, LHD               /SET PNTR TO 'FROM' LOC
LLE
LAM                     /FETCH CHAR TO TRANSFER
LHB                     /SET PNTR TO 'TO' LOC
LLC
LMA                     /STORE CHAR IN NEW LOC
RET                     /RET TO CALLING PGM
/
NCR, INC                /INCR 'TO' LO ADDR
JFZ NCRD                /IF NOT 0, SKIP NEXT
INB                     /INCR 'TO' PAGE
NCRD, INE               /INCR 'FROM' LO ADDR
RFZ                     /IF NOT 0, RET
IND                     /INCR 'FROM' PAGE
RET                     /RET TO CALLING PGM
/
DECR, SUI 001           /DECR VALUE IN ACC
LMA                     /STORE VALUE BACK IN MEM
RFC                     /RET IF NO CARRY
INL                     /ELSE, SET PNTR TO NX LOC
LAM                     /FETCH 2ND HALF OF VALUE
SUI 001                 /DECR 2ND HALF
LMA                     /STORE VALUE BACK IN MEM
RET                     /RET TO CALLING PGM
/
INCR, ADI 001           /INCR CONTENTS OF MEM LOC
LMA                     /RESTORE MEM LOC
RFC                     /IF NO CARRY, RETURN
INL                     /ELSE, FETCH NEXT LOC
LAM
ADI 001                 /INCR MEM LOC
LMA                     /RESTORE MEM LOC
RET                     /AND RETURN
/
```

```
    MNEMONIC                COMMENTS
 --------------          --------------

SV140, LLI 140          /SET PNTR TO LOC 140 PG 00
LHI 000
SVBE, LMB               /SAVE REG B THRU REG E
INL                     /IN MEMORY
LMC
INL
LMD
INL
LME
RET                     /RETURN
/
```

THE NEXT TWO UTILITY SUBROUTINES PRESENTED BELOW ARE USED TO TAKE CARE OF OUTPUTTING MESSAGES TO THE DISPLAY DEVICE. THE SUBROUTINE LABELED "HDLN" SETS UP A POINTER TO A COMBINATION OF A "CARRIAGE-RETURN" AND A "LINE-FEED" (STORED AS A "MESSAGE" OR CHARACTER STRING ON PAGE 00) AND THEN CALLS THE "MSG" SUBROUTINE TO OUTPUT THE "CR-LF" COMBINATION. THE SUBROUTINE "MSG" THE CAREFUL READER MAY OBSERVE, WILL SEND A STRING OF CHARACTERS TO THE DISPLAY OUTPUT SUBROUTINE STARTING WITH THE MEMORY ADDRESS POINTED TO BY THE CPU "H" & "L" REGISTERS WHEN THE "MSG" SUBROUTINE IS ENTERED. AFTER EACH CHARACTER IS SENT, THE "H" & "L" POINTER REGISTERS ARE ADVANCED AND THE PROCESS CONTINUED UNTIL A "ZERO" BYTE IS DETECTED. (REMEMBER THAT IT WAS MENTIONED EARLIER THAT TEXT STRINGS IN THE TEXT BUFFER WOULD BE TERMINATED BY A ZERO? "MSG" WILL ALSO BE USED TO OUTPUT TEXT FROM THE TEXT BUFFER IN ADDITION TO IT'S USE TO DISPLAY "FIXED" MESSAGES OF THE TYPE BEING DISCUSSED HERE!)

```
    MNEMONIC                COMMENTS
 --------------          --------------

HDLN, LLI 134           /SET PNTR TO C/R,L/F MSG
LHI 000
CAL MSG                 /PRINT THE C/R,L/F
LHI 000                 /RESET PAGE PNTR
RET                     /RETURN
/
MSG, LAM                /FETCH CHAR TO PRINT
NDA                     /=000?
RTZ                     /YES,DONE. RETURN
CAL PRINT               /PRINT THE CHAR.
CAL INMEM               /INCR MEM PNTR
JMP MSG                 /CONTINUE OUTPUT
```

AS THE READER NOW REALIZES, THE EDITOR PROGRAM IS CONTROLLED BY OPERATOR INPUTS WHICH SPECIFY A PARTICULAR COMMAND. WHEN A PARTICULAR COMMAND IS RECOGNIZED BY THE PROGRAM, OPERATION IS DIRECTED TO A MAJOR SUBROUTINE THAT PERFORMS THE REQUIRED FUNCTION. WHEN A SPECIFIC COMMAND SUBROUTINE IS ENTERED, IT MAY BE NECESSARY FOR THE COMMAND ROUTINE TO OBTAIN FURTHER INFORMATION FROM THE INPUT BUFFER TO DETERMINE THE LINE NUMBERS INVOLVED. THE FOLLOWING SUBROUTINE, LABELED "LDHILO," PERFORMS THE FUNCTIONS CONNECTED WITH DETERMINING THE LINE NUMBERS THAT WILL BE INVOLVED WITH THE COMMAND. THE SUBROUTINE FIRST READS IN THE LINE NUMBERS FROM THE LINE INPUT BUFFER (PLACED THERE WHEN THE OPERATOR ENTERS THE COMMAND) AND CHECKS EACH CHARACTER TO DETERMINE IF IT IS A

VALID OCTAL DIGIT. (IN THE RANGE 0 - 7, NUMBERS 8 AND 9 ARE NOT VALID!) IT THEN CONVERTS VALID DIGITS INTO AN 8 BIT BINARY VALUE AND STORES THE VALUES IN A SMALL TABLE ON PAGE 00. IF AN INVALID CONDITION IS DETECTED, AN "ERROR" DISPLAY ROUTINE IS EXECUTED. THE NUMBERS PROCESSED REPRESENT LINE NUMBERS (LOW LINE NUMBER, THEN HIGH LINE NUMBER) ASSOCIATED WITH THE COMMAND. THE "LDHILO" SUBROUTINE CONTINUES BY CHECKING TO SEE IF THE HIGH LINE NUMBER ENTERED BY THE OPERATOR IS LESS THAN OR EQUAL TO THE "CURRENT" LINE NUMBER. (THE "CURRENT" LINE NUMBER BEING INDICATED BY THE "LINE NUMBER COUNTER" AS THE HIGHEST LINE NUMBER CURRENTLY IN THE MAIN TEXT BUFFER STORAGE AREA.) THEN THE LOW LINE NUMBER ENTERED BY THE OPERATOR IS COMPARED TO THE HIGH LIMIT TO MAKE SURE IT IS LESS THAN OR EQUAL TO THAT VALUE. IF EITHER OF THESE TESTS FAIL, THE PROGRAM EXECUTES AN ERROR DISPLAY ROUTINE.

A LISTING FOR THE "LDHILO" SUBROUTINE IS PRESENTED NEXT FOLLOWED BY THE NUMBER HANDLING SUBROUTINES THAT IT UTILIZES.

MNEMONIC	COMMENTS
LDHILO, LLB	/SET PNTR TO S.A. OF INP BFR
INL	/MOVE PNTR TO 3RD CHAR
INL	
CAL OCTNM	/FETCH NMBR'S FROM INP BFR
LLI 166	/SET PNTR TO LO LIMIT
CAL LDBE	/FETCH LO & HI LIMIT
LAE	/COMPARE LO LIMIT TO HI LIMIT
CPC	
JTC ERR	/IF LO > HI, THEN ERROR
JFZ CKLI	/IF NOT =, THEN SKIP NEXT
LAD	/CHECK 2ND HALF LO > HI
CPB	
JTC ERR	/IF LO > HI, PRINT ERROR MSG
CKLI, LLI 163	/SET PNTR TO LINE NO.
LAM	/COMPARE HI LIMIT TO LINE NO.
CPE	
JTC ERR	/IF HI > LN NO., ERROR
RFZ	/RET IF NOT EQUAL
DCL	/CHECK 2ND HALF HI > LN NO.
LAM	
CPD	
JTC ERR	/IF HI > LN NO., ERROR
RET	/ELSE, RET TO CALLING PGM
/	
OCT, LLI 152	/SET PNTR TO 3RD DIGIT
LAM	
CPI 004	/IS DIGIT > 3?
RFS	/YES, RET WITH S FLAG RESET
NDI 003	/CLEAR CARRY
RRC	
RRC	/POSITION DIGIT
LBA	/SAVE IN REG B
DCL	/DECR PNTR
LAM	/FETCH NEXT DIGIT
RLC	
RLC	
RLC	/POSITION DIGIT
ADB	/ADD TO REG B

```
MNEMONIC                COMMENTS
--------------          --------------

LBA                     /SAVE IN REG B
DCL                     /DECR PNTR
LAM                     /FETCH LAST DIGIT
ADB                     /ADD TO REG B
LBA                     /STORE FINAL NO. IN REG B
LAI 200                 /SET S FLAG TO INDICATE
NDA                     /THAT THE NO. WAS VALID
RET                     /AND RETURN
/
OCTNM, LEL              /SAVE REG L
CAL OCTPR               /FETCH OCTAL NO. PAIR
LLI 166                 /SET STORAGE PNTR
LMB                     /STORE OCTAL PAIR IN LOC.166
INL                     /AND 167 ON PG 00
LMC
LLE
LAM                     /FETCH NEXT CHAR
CPI 254                 /COMMA?
JFZ SGL                 /NO, SINGLE INPUT
INL                     /YES, FETCH 2ND OCTAL PAIR
LEL
CAL OCTPR
SGL, LLI 170            /STORE OCTAL PAIR IN LOC. 170
LMB                     /AND 171 ON PG 00
INL
LMC
RET                     /RETURN
/
OCTPR, CAL DCDNM        /CONVERT INP TO OCT NO.
LCB                     /SAVE IN REG C
INE
JMP DCDNM               /CONVERT 2ND OCT NO. AND RET
/
DCDNM, CAL DCON         /FETCH OCTAL DIGITS
CAL OCT                 /FORM OCTAL NO. IN REG B
JFS ERR                 /IF INVALID, PRINT ERR MSG
RET                     /RETURN
/
DCON, LLI 150           /CLEAR DIGIT STORAGE TABLE
LMH                     /BY FILLING WITH 000
INL
LMH
INL
LMH
LLE
LOOP, CAL FNUM          /CHECK FOR VALID NO.
JTS CKLNH               /IF NOT, CHECK CHAR COUNT= 0
LAM
LDL
NDI 007                 /IF VALID, MASK OFF 260
LLI 150                 /STORE OCTAL NUMBER IN
LBM                     /TABLE AT LOC 150 PG 00
LMA                     /AND SHIFT OTHER NUMBERS
INL                     /UP THRU THE TABLE
LAM
LMB
INL
LMA
```

```
    MNEMONIC                COMMENTS
 --------------          --------------

LLD                     /RESTORE AND INCR INP PNTR
INL
JMP LOOP                /FETCH NEXT NUMBER
/
CKLNH, LAL
CPE                     /IS CHAR COUNT= 0?
JTZ ERR                 /YES, PRINT ERR MSG
LEL                     /NO, SET REG E AND RETURN
RET
/
FNUM, LAM               /IS CHAR A VALID NUMBER?
CPI 260
RTS                     /IF NOT, RET WITH S FLAG SET
SUI 270
ADI 200                 /IF SO, RET WITH S FLAG RESET
RET
/
LD140, LLI 140          /SET PNTR TO LOC 140 PG 00
LHI 000
LDBE, LBM               /LOAD REG B THRU REG E
INL                     /FROM MEMORY
LCM
INL
LDM
INL
LEM
RET                     /RETURN
/
```

WHEN A MAJOR EDITOR ROUTINE OPERATES TO ADD TEXT TO THE MAIN TEXT BUFFER, IT IS ADVISABLE TO MAKE SURE THAT THE ADDITION OF THE TEXT (FROM THE TEXT INPUT BUFFER) WILL NOT CAUSE THE TEXT BUFFER AREA TO "OVERFLOW" IT'S UPPER BOUNDARY. (THAT IS, ATTEMPT TO PLACE TEXT WHERE NO MEMORY EXIST!) THEREFORE, A SUBROUTINE IS REQUIRED TO CHECK FOR THE POSSIBILITY THAT THE MAIN TEXT BUFFER DOES NOT HAVE ENOUGH ROOM TO STORE THE NEW ADDITION. THE FOLLOWING SUBROUTINE, LABELED "CKOV" (CHECK OVERFLOW) IS USED FOR JUST THIS PURPOSE. THE "CKOV" SUBROUTINE FIRST CALCULATES THE NUMBER OF CHARACTERS IN THE LINE CURRENTLY BEING INPUTTED (AS IT RESIDES IN THE INPUT TEXT BUFFER) AND ADDS ONE TO THIS VALUE (FOR THE END OF LINE TERMINATOR BYTE). THE TOTAL VALUE IS THEN ADDED TO THE VALUE OF THE CURRENT TEXT BUFFER POINTER. IF THIS NEW VALUE SHOULD EXCEED THE VALUE OF THE HIGHEST ADDRESS ALLOWED FOR THE TEXT BUFFER (THE HIGHEST ADDRESS ALLOWED VALUE IS DETERMINED BY THE "KILL" ROUTINE TO BE PRESENTED LATER) THEN A "BUFFER FULL" INDICATING MESSAGE ROUTINE IS EXECUTED AND THE INFORMATION IN THE TEXT INPUT BUFFER IS NOT TRANSFERRED TO THE MAIN TEXT BUFFER. IN SUCH A CASE, THE PROGRAM WOULD BRANCH BACK TO THE EDITOR COMMAND MODE. IF THE NEW LINE WILL NOT CAUSE AN OVERFLOW CONDITION IN THE MAIN TEXT BUFFER, THE SUBROUTINE RETURNS TO THE CALLING PROGRAM TO ALLOW THE MAJOR ROUTINE TO CONTINUE NORMAL OPERATIONS.

```
    MNEMONIC                COMMENTS
 --------------          --------------

CKOV, LAE               /FETCH INP PNTR
CPB                     /IS INP A BLANK LINE?
JFZ NCHR                /NO, CONTINUE
INE                     /YES, SET OUP ONE SPACE
LAE
```

```
    MNEMONIC                COMMENTS
 --------------          --------------

NCHR, DCB               /DECR INP BFR LIMIT
SUB                     /CALC. CHAR COUNT+1
LLI 173                 /SET PNTR TO TBL AREA
LMA                     /SAVE CHAR COUNT+1
LCA                     /SAVE CHAR COUNT+1 IN REG C
LLI 160                 /SET PNTR TO TX BFR PNTR
INB                     /INCR INP BFR LIMIT
ADM                     /ADD CC+1 TO TX BFR PNTR
RFC                     /RET ON NO CARRY
INL                     /FETCH TX BFR PAGE PNTR
LAM
LLI 172                 /SET PNTR TO PAGE LIMIT
CPM                     /IS CUR PAGE = LAST AVAIL?
JTZ OFL                 /YES,PRINT OVERFLOW ERROR
LLI 160                 /NO, SET PNTR TO TX BFR PNTR
RET                     /RETURN TO CALLING PGM
/
OFL, CAL HDLN           /PRINT C/R, L/F
LAI 302                 /SET UP OVERFLOW ERROR CODE
JMP ERP                 /PRINT OVERFLOW ERROR
```

SOME OF THE MAJOR EDITOR ROUTINES PERFORM OPERATIONS THAT REQUIRE THAT A PARTICULAR LINE IN THE MAIN TEXT BUFFER BE LOCATED. THE NEXT SUBROUTINE TERMED "FND" SERVES TO LOCATE A PARTICULAR LINE IN THE TEXT BUFFER AND SAVE THE STARTING ADDRESS OF THAT LINE. THIS IS ACCOMPLISHED BY HAVING THE ROUTINE SCAN THE CONTENTS OF THE TEXT BUFFER AND COUNTING UP THE NUMBER OF "END OF LINE TERMINATORS" (ZERO BYTES) THAT IT FINDS UNTIL A PARTICULAR VALUE (CORRESPONDING TO A SPECIFIED LINE NUMBER) IS REACHED. AT THAT POINT, THE STARTING ADDRESS OF THE LINE IS SAVED IN CPU REGISTERS "D" AND "E." THE "FND" SUBROUTINE UTILIZES A SMALL SUB-ROUTINE ("ZLOK") TO ACTUALLY LOOK FOR A ZERO BYTE IN THE MAIN TEXT BUFF-ER. THE LISTINGS FOR BOTH "FND" AND "ZLOK" ARE SHOWN BELOW.

```
    MNEMONIC                COMMENTS
 --------------          --------------

FND, LEI 004            /SET START OF TX BFR PNTR
LDI 007
LLI 164                 /SET PNTR TO LINE COUNT
LHI 000
LMI 001                 /SET LINE COUNT TO 000 001
INL
LMH
FD1, LLI 167            /SET PNTR TO LO LIMIT
LAM                     /FETCH LO LIMIT
LLI 165                 /SET PNTR TO LINE COUNT
CPM                     /IS LINE COUNT = LO LIMIT?
JFZ FD2                 /NO,FIND START OF NEXT LINE
INL                     /CHECK OTHER HALF OF COUNT
LAM
DCL
DCL
CPM                     /IS LINE COUNT = LO LIMIT
RTZ                     /YES, RET TO CALLING PGM
```

```
    MNEMONIC                COMMENTS
 --------------          --------------

FD2, CAL ZLOK          /NO, SEARCH FOR NEXT LINE
LHI 000                /SET PNTR TO LINE COUNT
LLI 164
LAM                    /FETCH LINE COUNT
CAL INCR               /INCR LINE COUNT
JMP FD1                /CONTINUE
/
ZLOK, LHD              /SET PNTR TO TX BFR
LLE
ZL1, LAM               /FETCH CHAR FROM TX BFR
CAL INMEM              /INCR TEXT BFR PNTR
NDA                    /IS CHAR = 000?
JFZ ZL1                /NO, TRY NEXT CHAR
LEL                    /YES, SAVE TX BFR PNTR
LDH
RET                    /RETURN TO CALLING PGM
```

THE NEXT UTILITY SUBROUTINE IS SIMPLY USED TO FETCH THE LOW PORTION OF THE TEXT BUFFER LINE COUNTER. THE ROUTINE ALSO CONDITIONS THE FLAGS AFTER THE LOAD OPERATION. THIS SUBROUTINE IS FREQUENTLY USED BY MAJOR SUBROUTINES AS WILL BECOME APPARENT LATER.

```
    MNEMONIC                COMMENTS
 --------------          --------------

FLINO, LLI 162         /SET PNTR TO LINE NO.
LHI 000
LAM                    /FETCH LINE NO.
NDA                    /SET UP FLAGS
RET                    /RET TO CALLING PGM
```

THE FINAL GENERAL UTILITY SUBROUTINE TO BE PRESENTED IN THIS SECTION IS USED TO DETERMINE WHETHER A LINE NUMBER SPECIFIED BY AN OPERATOR HAS THE INVALID VALUE OF ZERO. THIS SUBROUTINE IS THUS USED TO INDICATE THAT AN ERROR CONDITION IS PRESENT IN THE COMMAND SEQUENCE ENTERED BY THE OPERATOR.

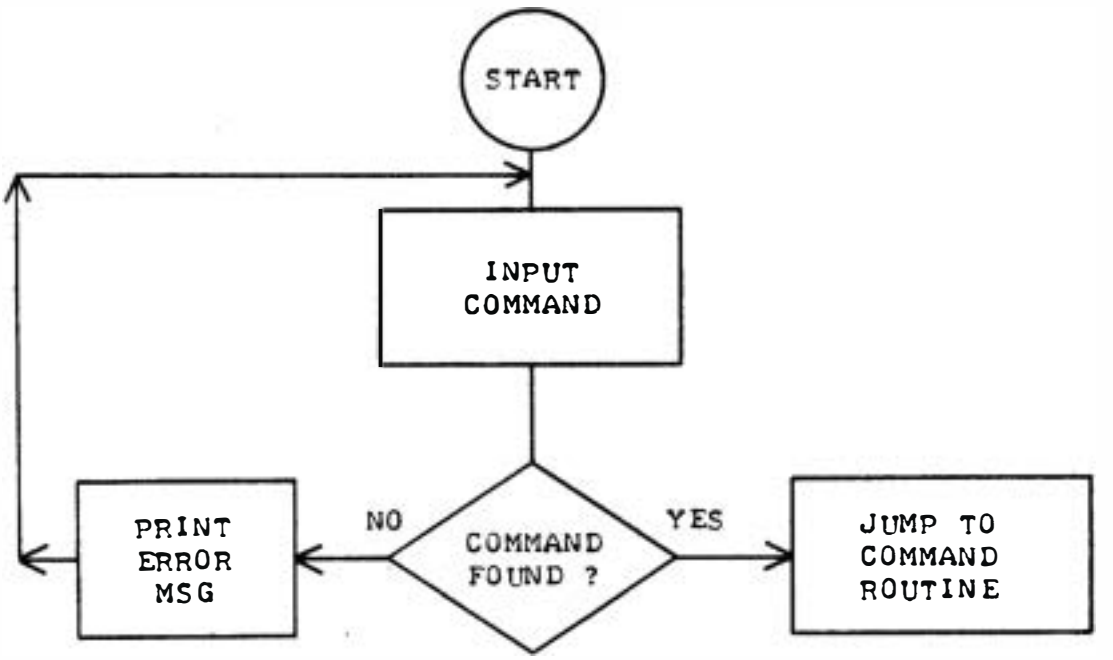
```
    MNEMONIC                COMMENTS
 --------------          --------------

FLO, LLI 166           /SET PNTR TO LO LIMIT
LHI 000
LAM                    /FETCH LO LIMIT
NDA                    /IS 1ST HALF = 000?
RFZ                    /NO,RET WITH Z FLAG RESET
INL                    /YES, FETCH 2ND HALF
LAM
NDA                    /SET UP Z FLAG
RET                    /RET TO CALLING PGM
/
```

MAJOR ROUTINES FOR THE EDITOR PROGRAM

"COMMANDS" INPUT ROUTINE

THIS SECTION DESCRIBES THE MAJOR OPERATING ROUTINES USED IN THE DESCRIBED EDITOR PROGRAM. THE FIRST SUCH ROUTINE IN THIS CATEGORY IS DESIGNATED THE "COMMAND INPUT ROUTINE" AND THE START OF THE ROUTINE HAS BEEN GIVEN THE LABEL "INCMD." THE COMMAND INPUT ROUTINE UTILIZED BY THIS EDITOR PROGRAM HAS A VERY GENERAL FORMAT WHICH CAN BE APPLIED TO MANY OTHER TYPES OF PROGRAMS. THE ROUTINE ESSENTIALLY "INTERPRETS" COMMANDS ISSUED BY THE OPERATOR ON AN INPUT DEVICE AND JUMPS TO AN ADDRESS ASSOCIATED WITH THE COMMAND IT RECEIVES. THE BASIC OPERATING PORTION OF THIS ROUTINE IS THE SAME REGARDLESS OF HOW MANY DIFFERENT TYPES OF COMMANDS ONE MIGHT WISH TO DEFINE. TO USE THE ROUTINE FOR OTHER TYPES OF PROGRAMS (OR TO EXTEND THE CAPABILITY OF THE EDITOR PROGRAM ITSELF), ONE NEED MERELY LENGTHEN THE "COMMAND LOOK UP TABLE" AND CHANGE THE VALUE IN A COUNTER INITIALIZING INSTRUCTION THAT INDICATES HOW MANY BYTES ARE OCCUPIED BY THE LOOK UP TABLE.

THE BASIC FLOW CHART FOR THE COMMANDS INPUT ROUTINE IS ILLUSTRATED BELOW. THE FLOW CHART SHOWS CLEARLY THE ACTUAL SIMPLICITY BEHIND THE CONCEPT UTILIZED BY THE ROUTINE.



FLOW CHART - COMMANDS INPUT ROUTINE

THE COMMAND INPUT ROUTINE STARTS BY FIRST DISPLAYING A "COMMAND MODE" SYMBOL ON THE DISPLAY DEVICE. THIS SYMBOL (DEFINED AS A ">" MARK) INDICATES TO THE OPERATOR THAT THE EDITOR PROGRAM IS CURRENTLY IN THE COMMAND MODE. THE ROUTINE THEN CALLS UPON THE OPERATOR INPUT SUBROUTINE (TO BE DESCRIBED SHORTLY) WHICH WILL WAIT FOR THE OPERATOR TO ENTER A COMMAND, AND WHEN THIS OCCURS, STORE IT IN THE TEXT INPUT BUFFER. ONCE A COMMAND STRING HAS BEEN ENTERED, THE COMMAND INPUT ROUTINE OPERATES AS FOLLOWS. THE FIRST CHARACTER IN THE TEXT INPUT BUFFER IS ASSUMED TO BE

THE COMMAND INDICATING CHARACTER. THE ROUTINE OBTAINS THE FIRST CHARACTER IN THE INPUT TEXT BUFFER. IT THEN COMPARES THIS CHARACTER WITH EVERY THIRD BYTE IN A "COMMAND LOOK UP TABLE." THE "COMMAND LOOK UP TABLE" HAS THE FORMAT ILLUSTRATED HERE:

```
BYTE #N    XXX   =   ASCII CODE FOR A COMMAND CHARACTER
BYTE N+1   YYY   =   LOW ADDR OF ASSOC COMMAND ROUTINE
BYTE N+2   ZZZ   =   PAGE ADDR OF ASSOC COMMAND ROUTINE
BYTE N+3   MMM   =   ASCII CODE FOR A COMMAND CHARACTER
BYTE N+4   NNN   =   LOW ADDR OF ASSOC COMMAND ROUTINE
BYTE N+5   000   =   PAGE ADDR OF ASSOC COMMAND ROUTINE
BYTE N+6   AAA   =   ASCII CODE FOR A COMMAND CHARACTER
                     .
                     .
                     .
REPEAT SEQUENCE TO END OF COMMAND LOOK UP TABLE
```

IF A MATCH IS FOUND BETWEEN THE CHARACTER INPUTTED AS THE COMMAND AND AN ENTRY IN THE COMMAND TABLE, THE ADDRESS IN THE NEXT TWO BYTES OF THE COMMAND LOOK UP TABLE IS OBTAINED AND TRANSFERRED TO TWO SPECIAL LOCATIONS ON PAGE 00 OF THE EDITOR PROGRAM. THESE TWO LOCATIONS FORM THE SECOND AND THIRD BYTES (ADDRESS PORTION) OF A "JMP" (JUMP) INSTRUCTION WHICH IS THEN EXECUTED TO DIRECT THE OPERATION OF THE EDITOR PROGRAM TO THE SPECIFIC COMMAND SUBROUTINE THAT THE OPERATOR DESIRED WHEN THE COMMAND CHARACTER WAS ENTERED! IF, HOWEVER, THERE IS NO MATCHING ENTRY BETWEEN WHAT THE OPERATOR ENTERED, AND THE CHARACTERS REPRESENTING COMMANDS IN THE COMMAND LOOK UP TABLE, THEN AN ERRONEOUS OPERATOR ENTRY IS ASSUMED AND THE ROUTINE EXECUTES AN ERROR DISPLAY ROUTINE.

A LISTING OF THE "COMMAND LOOK UP TABLE" FOLLOWED BY THE ACTUAL "INCMD" ROUTINE USED IN THE EDITOR PROGRAM BEING PRESENTED IS SHOWN BELOW. FOR THE LISTING SHOWN, THE COMMAND LOOK UP TABLE IS ASSUMED TO BE STORED ON PAGE 01 STARTING AT LOCATION 000 AND TO CONTAIN ENTRIES TO HANDLE NINE DIFFERENT TYPES OF COMMANDS.

```
    MNEMONIC                COMMENTS
 --------------          --------------
/
/COMMAND LOOK UP TABLE
/
301                 /APPEND
200
002
303                 /CHANGE
164
004
304                 /DELETE
015
003
311                 /INSERT
337
003
313                 /KILL
350
002
314                 /LIST
140
003
```

```
   MNEMONIC                COMMENTS
-------------          --------------

322                    /READ
275
002
323                    /SEARCH
175
004
327                    /WRITE
321
002
/
INCMD, LHI 000         /SET PNTR TO C/R,L/F,>
LLI 130
CAL MSG                /PRINT COMMAND HEADING
CAL CDIN               /INPUT THE COMMAND
LAM                    /FETCH THE 1ST CHAR
LDI 011                /SET NO. OF CMND CNTR
LLH                    /SET CMND TABLE PNTR
LHI 001
LKCMD, CPM             /IS INP = CMND CHAR?
JTZ FOUND              /YES,PROCESS CMND
INL                    /NO, ADV. CMND TBL PNTR
INL
INL
DCD                    /LAST CMND CHAR CHECKED?
JFZ LKCMD              /NO, TRY AGAIN
/                       YES, PRINT ERR MSG
ERR, CAL HDLN          /PRINT C/R,L/F
LAI 311                /SET LETTER 'I' FOR OUTPUT
ERP, CAL PRINT         /PRINT ERR MSG
JMP INCMD              /RETURN TO CMND MODE WITHOUT
/
FOUND, INL             /ADV CMND PNTR TO ADDR OF CMND
LAM                    /FETCH LO ADDR OF CMND
INL
LDM                    /FETCH PG ADDR OF CMND
LLI 156                /SET PNTR TO JUMP INSTRUCTION
LHI 000
LMA                    /SET LO ADDR OF CMND
INL
LMD                    /SET PG ADDR OF CMND
LLE                    /SET PNTR TO END OF CMND INP
JMP 155 000            /JUMP TO THE CMND ROUTINE
/
```

INPUT (LINE BUFFER) ROUTINE

THE INPUT BUFFER ROUTINE HAS BEEN WRITTEN IN A FLEXIBLE FASHION TO ALLOW IT TO SERVE ALL THE EDITOR ROUTINES WHICH REQUIRE INPUT FROM THE OPERATOR INPUT DEVICE. (THE ROUTINE CAN BE OF GENERAL USE IN OTHER TYPES OF PROGRAMS WHICH REQUIRE THE INPUTTING OF A LINE OF INFORMATION FROM AN EXTERNAL DEVICE INTO A BUFFER AREA IN MEMORY). THE ROUTINE ACCEPTS INPUTS FROM AN EXTERNAL DEVICE (VIA A USER DEFINED " RCV " ROUTINE) AND STORES THE CHARACTERS IT RECEIVES IN A BUFFER (MENTIONED PREVIOUSLY AS RESIDING ON PAGE 00) UNTIL A LINE TERMINATOR CHARACTER IS RECEIVED. THE ROUTINE HAS BEEN PROVIDED WITH CAPABILITIES THAT ENABLE AN OPERATOR TO MAKE CORRECTIONS BY DELETING INDIVIDUAL CHARACTERS AFTER

THEY HAVE BEEN ENTERED ON THE OPERATOR'S INPUT DEVICE, OR EVEN "ERASE" THE ENTIRE LINE BEING INPUTTED, AS LONG AS SUCH CORRECTIONS ARE MADE BEFORE A LINE TERMINATING CHARACTER HAS BEEN PROCESSED.

A GENERAL FLOW CHART OF THE INPUT ROUTINE IS SHOWN ON THE FOLLOWING PAGE. THE READER MAY REFER TO THIS DIAGRAM DURING THE FOLLOWING DISCUSSION.

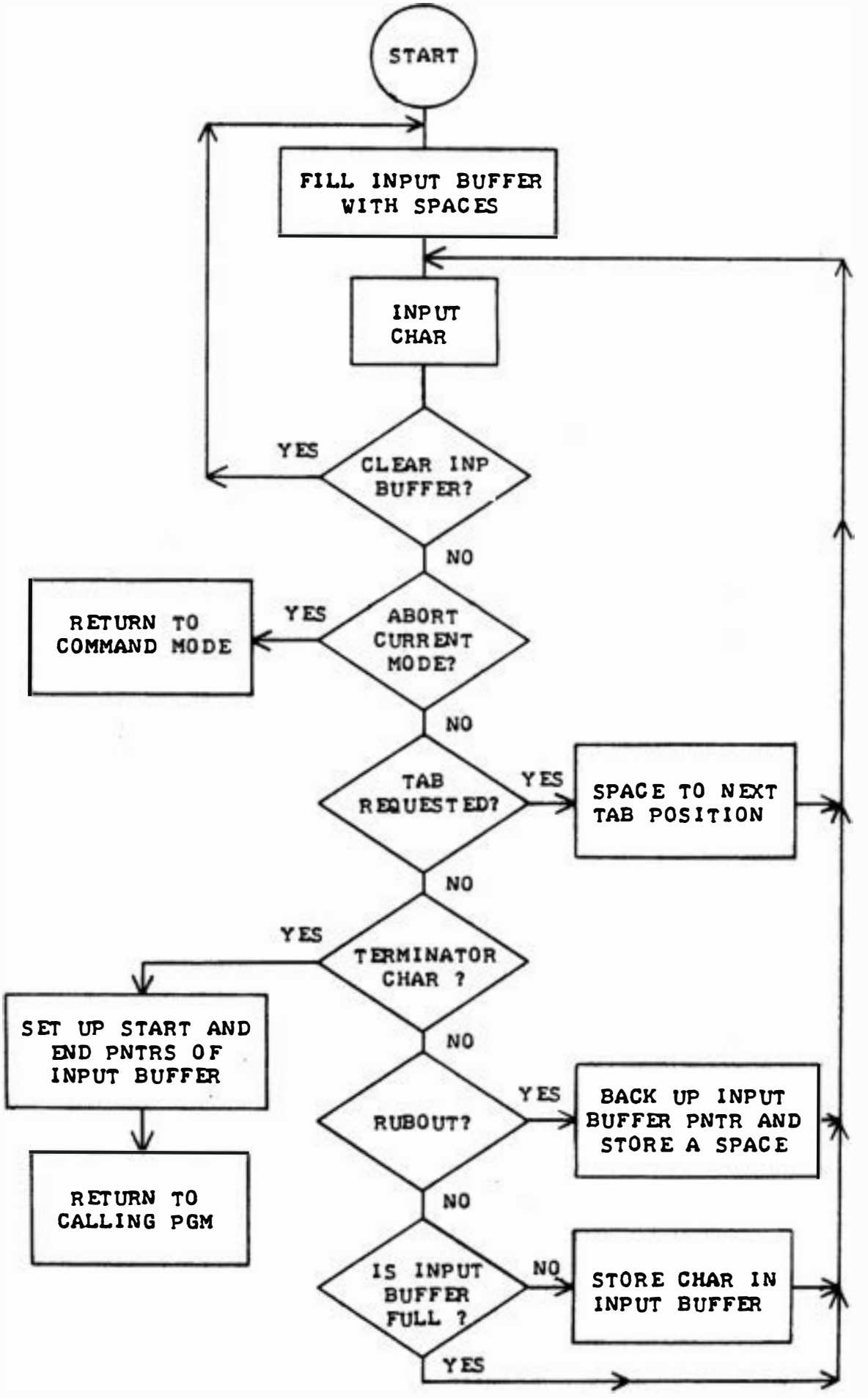
THE FIRST OPERATION PERFORMED BY THIS ROUTINE IS TO FILL THE INPUT LINE BUFFER WITH THE ASCII CODE FOR "SPACES." THIS PROCESS ESSENTIALLY "CLEARS OUT" THE INPUT BUFFER AREA. THE STARTING ADDRESS OF THE INPUT BUFFER IS SET UP BY CALLING THE "FBFLM" SUBROUTINE. "FBFLM," AS PREVIOUSLY DETAILED, STORES THE STARTING ADDRESS OF THE INPUT BUFFER IN CPU REGISTER "B" AND EFFECTIVELY DETERMINES THE LENGTH OF THE INPUT BUFFER. THE LINE INPUT ROUTINE THEN CALLS THE USER DEFINED "RCV" SUBROUTINE WHICH ACCEPTS CHARACTERS FROM THE OPERATOR'S INPUT DEVICE. THE "RCV" SUBROUTINE RETURNS TO THE LINE INPUT ROUTINE WITH THE ASCII CODE FOR THE CHARACTER IT HAS RECEIVED IN THE ACCUMULATOR. THE ROUTINE THEN PROCEEDS TO TEST THE CHARACTER IT HAS JUST RECEIVED TO SEE IF IT IS ONE OF A NUMBER OF SPECIAL "CONTROL" CHARACTERS.

THE FIRST SUCH "CONTROL" CHARACTER TESTED FOR IS DEFINED AS "CONTROL S." "CONTROL S" IS TYPICALLY GENERATED ON AN ASCII ENCODED KEYBOARD DEVICE BY SIMULTANEOUSLY DEPRESSING THE "CTRL" KEY AND THE KEY FOR THE LETTER "S." THE ASCII CODE FOR THIS COMBINATION IS "223" OCTAL. THE RECEIPT OF THIS CODE BY THE INPUT ROUTINE INDICATES THAT THE OPERATOR DESIRES TO ERASE THE CURRENT CONTENTS OF THE LINE INPUT BUFFER (BECAUSE, FOR INSTANCE, THE OPERATOR HAS MADE A MISTAKE NEAR THE BEGINNING OF THE CURRENT LINE). IF THIS CHARACTER IS DETECTED, THE INPUT ROUTINE WILL PRESENT A CARRIAGE RETURN AND LINE FEED COMBINATION TO THE OUTPUT DEVICE USING THE UTILITY SUBROUTINE "HDLN." THE ROUTINE THEN JUMPS BACK TO THE START OF THE INPUT ROUTINE TO CLEAR THE INPUT BUFFER AREA AND ALLOW THE OPERATOR TO START THE INPUT PROCESS AGAIN.

IF THE CHARACTER RECEIVED IS NOT A "CONTROL S," THE ROUTINE NEXT TESTS FOR A "CONTROL D" (ASCII CODE 204 OCTAL). RECEIPT OF A "CONTROL D" INDICATES THE OPERATOR DESIRES TO ABORT THE CURRENT INPUT OPERATION AND RETURN TO THE "COMMAND" MODE OF THE EDITOR PROGRAM. THE "CONTROL D" CHARACTER IS TYPICALLY USED TO TERMINATE THE "TEXT INPUT MODE" WHEN THE OPERATOR HAS FINISHED ENTERING TEXT UNDER AN "APPEND," "CHANGE," "INSERT" OR "SEARCH" OPERATION. OR, IF THE OPERATOR DESIRES TO COMPLETELY NEGATE A "COMMAND" INPUT STRING.

THE NEXT CHARACTER THE INPUT ROUTINE TESTS FOR IS A "CONTROL I." THIS CHARACTER HAS AN ASCII CODE OF 211 (OCTAL) AND IS ALSO COMMONLY REFERRED TO AS THE ASCII CODE FOR A "TAB" FUNCTION. RECEIPT OF THIS CHARACTER BY THE INPUT ROUTINE WILL CAUSE A "TAB" FUNCTION TO OCCUR! THAT IS, WHEN THE "TAB" CODE IS DETECTED, THE PROGRAM WILL INSERT SPACES IN THE TEXT INPUT BUFFER UNTIL THE NEXT "TAB" POSITION IN THE BUFFER IS REACHED. THIS VERSION OF THE EDITOR PROGRAM HAS BEEN DESIGNED TO PROVIDE A "TAB" MARKER AT EVERY EIGHT (DECIMAL) LOCATIONS IN A LINE. FOR EXAMPLE, IF THE "TAB" CHARACTER WAS DETECTED AFTER THREE CHARACTERS HAD BEEN PLACED IN THE INPUT TEXT BUFFER, THE INPUT ROUTINE WOULD INSERT FIVE SPACES IN THE BUFFER (TO REACH THE NEXT TAB MARKER POSITION) AND ALSO SEND FIVE "SPACING" CHARACTERS TO THE DISPLAY DEVICE. THE "TAB" FUNCTION IN AN EDITOR PROGRAM MAKES IT EASY FOR AN OPERATOR TO PREPARE NEATLY COLUMNED ENTRIES TO SEPARATE PORTIONS OF DATA. A FREQUENT USE FOR SUCH A FUNCTION IS TO SEPARATE, FOR INSTANCE, LABEL, MNEMONIC, OPERATOR, AND COMMENTS FIELDS IN A SOURCE LISTING FOR A PROGRAM LISTING!

THE NEXT CHARACTER CHECKED FOR IS A "LINE TERMINATOR." THE INPUT



TEXT INPUT BUFFER SUBROUTINE FLOW CHART

ROUTINE IN THIS EDITOR HAS BEEN DESIGNED SO THAT ANY CHARACTER HAVING AN ASCII CODE OF 215 (OCTAL) OR LESS (OTHER THAN THE THREE "CONTROL" CODES PREVIOUSLY MENTIONED) MAY SERVE AS A LINE TERMINATOR. THE ASCII CODE FOR "CARRIAGE RETURN" (215) IS GENERALLY RECOMMENDED AS THE TERMINATING CHARACTER FOR MOST OPERATIONS WITH THE DESCRIBED EDITOR PROGRAM AS THIS CHARACTER WILL CAUSE THE DISPLAY DEVICE TO RETURN TO THE LEFT HAND SIDE OF THE DISPLAY. HOWEVER, AT CERTAIN POINTS, SUCH AS WHEN UTILIZING THE "SEARCH" FUNCTION IN THE EDITOR PROGRAM, IT MAY NOT BE DESIRABLE TO HAVE THE DISPLAY PERFORM A CARRIAGE RETURN (AT THE CONCLUSION OF SPECIFYING THE "SEARCH" COMMAND, BUT BEFORE THE "SEARCH CHARACTER" HAS BEEN DESIGNATED). IN SUCH CASES, THE OPERATOR MAY USE A CODE SUCH AS "CONTROL L" (214 OCTAL IN ASCII CODE) TO SERVE AS A LINE TERMINATOR. DOING SO WILL CAUSE THE ROUTINE TO EXIT THE INPUT ROUTINE WITHOUT DISPLAYING A CARRIAGE RETURN (OR OTHER CHARACTER SINCE A "CONTROL L" IS GENERALLY USED FOR DEFINING A NON-DISPLAYING FUNCTION).

THE FINAL "CONTROL" CHARACTER TESTED FOR BY THE INPUT ROUTINE IS THE CODE (377 OCTAL) ASSIGNED TO THE "RUBOUT" FUNCTION. RECEIPT OF THIS CODE INDICATES TO THE INPUT ROUTINE THAT THE PREVIOUS CHARACTER ENTERED BY THE OPERATOR IS TO BE DELETED FROM THE INPUT BUFFER. THIS IS ACCOMPLISHED BY BACKING UP THE INPUT BUFFER POINTER ONE LOCATION AND INSERTING THE CODE FOR A "SPACE" TO EFFECTIVELY "ERASE" THE PREVIOUS CHARACTER ENTRY FROM THE INPUT BUFFER. AN OPERATOR MAY ERASE MORE THAN ONE CHARACTER IN THE INPUT BUFFER BY USING THE "RUBOUT" FUNCTION SEVERAL TIMES IN SUCCESSION.

IF NONE OF THE PREVIOUSLY MENTIONED "CONTROL" CHARACTERS ARE FOUND BY THE INPUT ROUTINE, THE CODE FOR THE CHARACTER ENTERED WILL BE STORED IN THE INPUT BUFFER AND THE INPUT BUFFER POINTER WILL BE ADVANCED. THIS PROCESS WILL CONTINUE AS LONG AS CHARACTERS ARE ENTERED FROM THE OPERATOR INPUT DEVICE. HOWEVER, ONCE THE INPUT BUFFER IS FILLED, THE INPUT BUFFER POINTER WILL NOT BE ADVANCED. (THIS FEATURE PREVENTS AN OPERATOR FROM INADVERTANTLY CAUSING THE INPUT BUFFER TO OVERFLOW INTO THE OPERATING PORTION OF THE EDITOR PROGRAM!)

IT SHOULD BE EASY TO SEE THAT THE READER MAY ELECT TO ASSIGN DIFFERENT CHARACTERS TO OPERATE AS "CONTROL" CHARACTERS IN THE INPUT ROUTINE. THIS MAY READILY BE ACCOMPLISHED BY CHANGING THE IMMEDIATE PORTION OF THE "CPI" INSTRUCTIONS IN THE INPUT ROUTINE. FOR EXAMPLE, IF A USER DESIRED TO HAVE THE CODE FOR "CONTROL O" (217) SERVE TO "ERASE" ONE CHARACTER INSTEAD OF THE "RUBOUT" CODE SHOWN IN THE ROUTINE, THE USER COULD SIMPLY SUBSTITUTE "217" FOR "377" IN THE "CPI" INSTRUCTION USED TO TEST FOR THE "RUBOUT."

ADDITIONALLY, IF A USER DESIRED TO ADD OTHER TYPES OF "CONTROL" FUNCTIONS TO THE INPUT ROUTINE, IT COULD BE READILY DONE BY ADDING ADDITIONAL "CPI" INSTRUCTIONS AND APPROPRIATE "JUMPS" TO USER PROVIDED ROUTINES THAT WOULD ACCOMPLISH THE DESIRED SERVICES.

THE DETAILS OF THE INPUT BUFFER ROUTINE ARE SHOWN BY THE LISTING PROVIDED BELOW. THE ROUTINE IS REFERRED TO BY THE LABEL "CDIN."

```
    MNEMONIC                COMMENTS
 ---------------         ---------------

BDLN, CAL HDLN          /PRINT C/R,L/F
CDIN, CAL FBFLM         /FETCH INP BFR LIMIT
LLB                     /SET INP BFR PNTR
LAI 240                 /FILL INP BFR WITH SPACES
```

```
MNEMONIC                COMMENTS
--------------          --------------

SP1, LMA
INL                     /IS ENTIRE BFR FULL?
JFZ SP1                 /NO, CONTINUE
LLB                     /YES, SET INP BFR PNTR
IN2, CAL RCV            /INP A CHAR
CPI 223                 /INP = CNT'L S?
JTZ BDLN                /YES, DELETE LINE INPUT
CPI 204                 /INP = CNT'L D?
JTZ INCMD               /YES, RET TO CMND MODE
CPI 211                 /INP = TAB?
JTZ TAB                 /YES, DO TAB FUNCTION
CPI 216                 /INP = CNT'L CHAR < M?
JTC RTN2                /YES, RET TO CALLING PGM
CPI 377                 /INP = RUBOUT?
JTZ BDCR                /YES, DELETE CHAR
INL                     /IS INP BFR FULL?
DCL
JTZ IN2                 /YES. DON'T STORE CHAR
LMA                     /NO, STORE CHAR
INL                     /INCR INP BFR PNTR
JMP IN2                 /GET NEXT CHAR
/
RTN2, CAL FBFLM         /FETCH INP BFR LIMIT
LLB                     /RESET INP BFR PNTR
LAM                     /FETCH 1ST CHAR
RET                     /RETURN TO CALLING PGM
/
BDCR, CAL FBFLM         /FETCH INP BFR LIMIT
LAB
CPL                     /ANY CHAR'S STORED YET?
JTZ IN2                 /NO, GET NEXT CHAR
DCL                     /YES, BACK UP PNTR
LMI 240                 /STORE A SPACE
JMP IN2                 /GET NEXT CHAR
/
TAB, INL                /CHECK FOR INP BFR FULL
DCL
JTZ IN2                 /IF FULL, IGNORE TAB
LAI 240                 /SET UP SPACE CHAR IN ACC
LMA                     /STORE SPACE IN INP BFR
CAL PRINT               /PRINT SPACE
INL                     /INCR INP BFR PNTR
LAL
NDI 007                 /IS TAB OVER TO LIMIT
JTZ IN2                 /YES, INP NEXT CHAR
JMP TAB                 /NO, CONT. WITH SPACES
/
```

THE "KILL" COMMAND ROUTINE

THE "KILL" COMMAND ROUTINE IS USED WHEN AN OPERATOR HAS REQUESTED THE "KILL" (K) FUNCTION. THE "KILL" COMMAND SHOULD BE THE FIRST FUNTION AN OPERATOR HAS THE EDITOR PROGRAM PERFORM WHEN A SYSTEM IS INITIALIZED, AND IT IS ALSO USED WHENEVER ONE WANTS TO EFFECTIVELY "ERASE" THE MAIN TEXT STORAGE BUFFER.

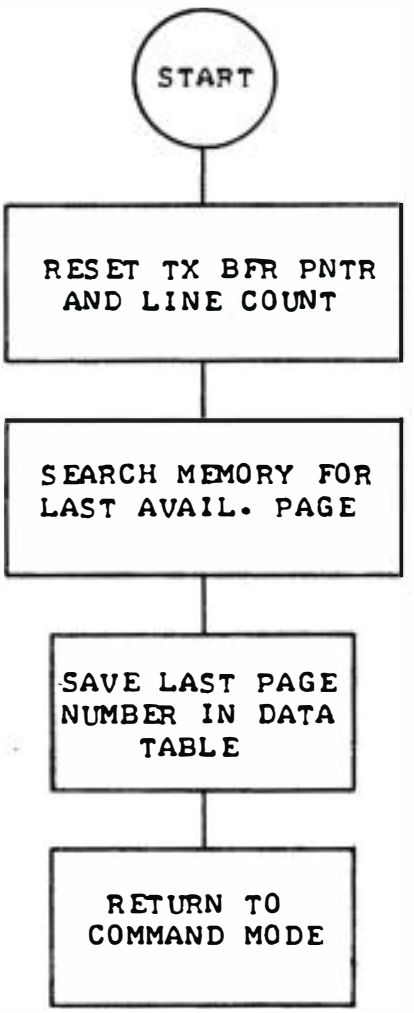
THE "KILL" SUBROUTINE RESETS THE TEXT BUFFER POINTER TO THE START OF THE MAIN TEXT BUFFER AREA AND ALSO RESETS THE LINE NUMBER COUNTER TO A VALUE OF ZERO. THESE TWO SIMPLY OPERATIONS EFFECTIVELY "WIPE OUT" ANY PREVIOUS CONTENTS OF THE TEXT BUFFER AS FAR AS THE EDITOR PROGRAM IS CONCERNED SINCE THE EDITOR PROGRAM USES THOSE TWO ESSENTIAL PIECES OF INFORMATION TO CONTROL STORAGE OF INFORMATION IN THE MAIN TEXT BUFFER. RESETTING THEIR VALUES IS A QUICK AND EASY WAY TO ACCOMPLISH THE OBJECTIVE. (EVEN THOUGH NO CHANGES IN THE TEXT BUFFER AREA ITSELF HAVE BEEN EFFECTED, THE INFORMATION THAT WAS IN THERE CAN BE CONSIDERED AS "LOST" SINCE THE EDITOR PROGRAM NO LONGER IS ABLE TO "ACCESS" SPECIFIC PIECES OF INFORMATION. THE METHOD OF RESETTING THE POINTER AND COUNTER IS JUST AS EXPEDIENT AS IF, SAY, ONE USED A ROUTINE TO FILL THE TEXT BUFFER STORAGE AREA WITH ZERO BYTES TO "ERASE" THE PREVIOUS CONTENTS.)

THE "KILL" SUBROUTINE ALSO DOES AN ADDITIONAL FUNCTION. IT HAS BEEN DESIGNED TO "SIZE UP" MEMORY IN ORDER TO DETERMINE THE UPPER ADDRESS LIMIT OF THE MAIN TEXT BUFFER STORAGE AREA. THIS PORTION OF THE ROUTINE UTILIZES THE ASSUMPTION THAT IN MOST MICROCOMPUTER SYSTEMS, IF ONE ATTEMPTS TO READ DATA FROM A MEMORY ADDRESS WHERE NO MEMORY HAS BEEN INSTALLED, ONE WILL OBTAIN ALL "ONES" OR 377 OCTAL. TO DETERMINE WHERE MEMORY ACTUALLY EXIST IN A SYSTEM (ASSUMING AVAILABLE MEMORY HAS BEEN INSTALLED IN THE LOWEST ADDRESS RANGE) ONE CAN SIMPLY HAVE THE COMPUTER PERFORM THE FOLLOWING OPERATIONS. FIRST, THE COMPUTER IS DIRECTED TO ATTEMPT TO WRITE ALL ZEROS TO A WORD IN MEMORY AT THE HIGHEST POSSIBLE PAGE ADDRESS FOR THE SYSTEM. IT THEN ATTEMPTS TO READ BACK FROM THE SAME LOCATION. IF THE SYSTEM PHYSICALLY HAS MEMORY ELEMENTS AT THAT ADDRESS, THE COMPUTER SHOULD OBTAIN THE ZERO BYTE THAT IT PREVIOUSLY WROTE TO THAT LOCATION! HOWEVER, IF NO PHYSICAL MEMORY IS PRESENT, THEN THE COMPUTER WOULD NOT OBTAIN THE ZERO VALUE. (REMEMBER THE ASSUMPTION MADE ABOVE. IF THE READER'S SYSTEM SHOULD BEHAVE DIFFERENTLY, SOME MODIFICATION TO THE ALGORITHM MIGHT BE REQUIRED BUT THE SAME CONCEPT WOULD QUITE LIKELY APPLY.) IF THE PROGRAM DOES NOT READ BACK WHAT IT WROTE AT THE HIGHEST POSSIBLE PAGE ADDRESS FOR THE SYSTEM, THEN THE PROGRAM SIMPLY DECREMENTS THE PAGE POINTER AND TRIES THE OPERATION AT AN ADDRESS ONE PAGE LOWER. THIS PROCESS CONTINUES UNTIL THE PROGRAM REACHES A POINT WHERE IT IS ABLE TO READ BACK THE ZERO BYTE THAT IT WROTE INTO A MEMORY LOCATION. THE VALUE OF THE MEMORY PAGE POINTER AT THAT POINT IS THEN SAVED, IN THE EDITOR PROGRAM, AS AN INDICATOR OF THE HIGHEST PAGE ADDRESS AVAILABLE FOR USE BY THE MAIN TEXT BUFFER IN THE SYSTEM IN WHICH THE PROGRAM IS BEING OPERATED! (THE TEXT BUFFER PAGE LIMIT IS USED BY A PREVIOUSLY ILLUSTRATED UTILITY SUBROUTINE LABELED "CKOV" TO DETERMINE IF INFORMATION ABOUT TO BE ADDED TO THE TEXT BUFFER WILL CAUSE THE MAIN TEXT BUFFER TO "OVERFLOW" BEYOND PHYSICAL MEMORY PRESENT IN THE SYSTEM.)

A LISTING OF THE "KILL" COMMAND ROUTINE IS SHOWN BELOW, FOLLOWED BY A FLOW DIAGRAM THAT SUMMARIZES IT'S OPERATIONS.

```
   MNEMONIC                COMMENTS
 -------------           -------------

KILL, LHI 000        /SET PNTR TO TABLE AREA
LLI 160
LMI 004              /SET TX BFR PNTR TO START
INL                  /OF THE TX BFR AREA
LMI 007
INL                  /SET PNTR TO LINE NO.
LMH                  /SET LINE NO. TO 000 000
INL
LMH
```

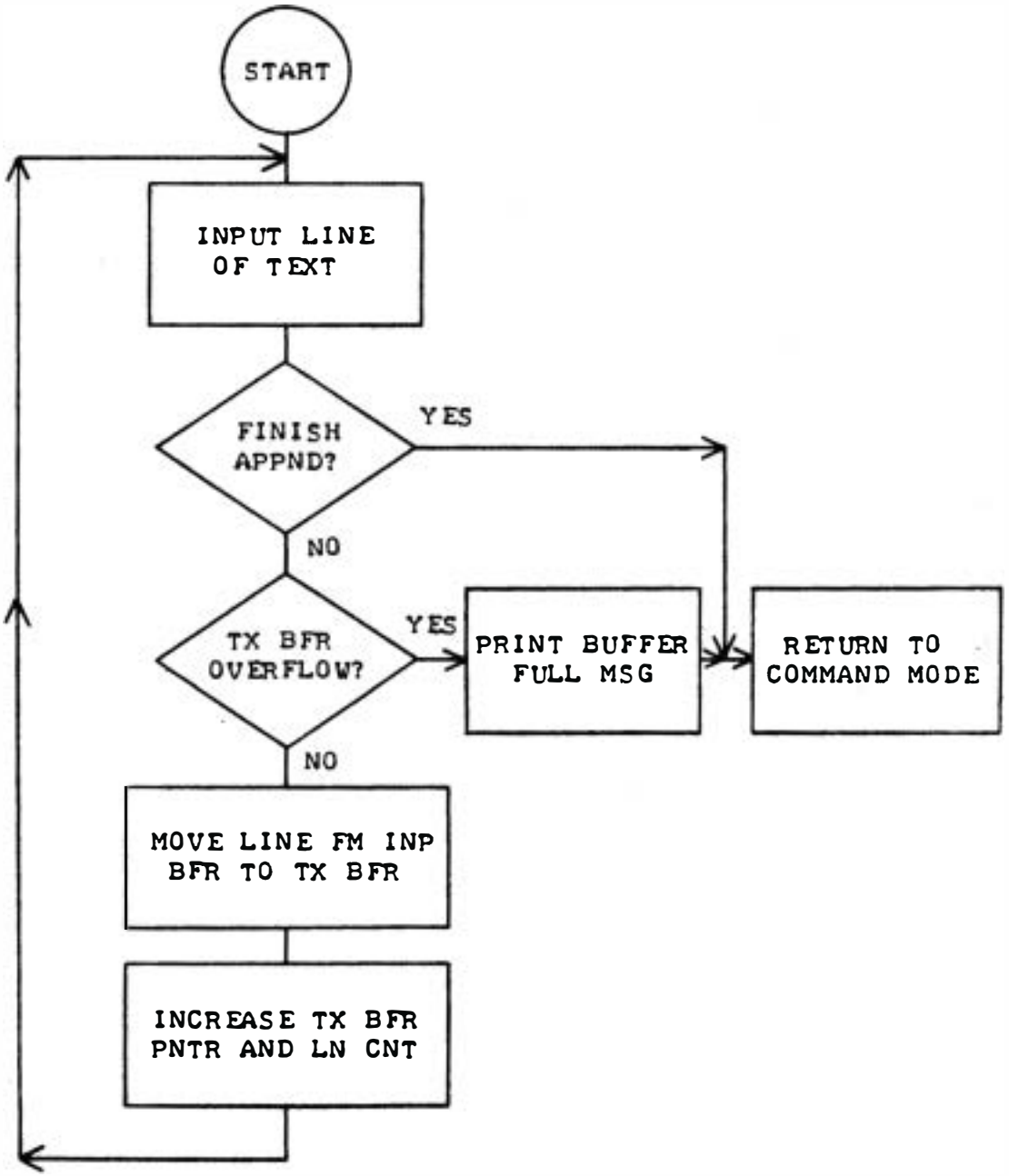
```
MNEMONIC                COMMENTS
--------------          --------------

LLH                     /SET PNTR TO THE LAST POSSIBLE
LHI 077                 /PAGE OF MEMORY
CKE, LBM                /SAVE MEM CONTENTS
LML                     /WRITE 000 TO MEM LOC
LAM                     /READ IT BACK
LMB                     /RESTORE MEM CONTENTS
NDA                     /IS MEMORY AVAILABLE?
JTZ ENM                 /YES, SET MAX PAGE LIMIT
DCH                     /NO, DECR PAGE PNTR
JMP CKE                 /CHECK NEXT PAGE
ENM, LBH                /SAVE PAGE NUMBER
LLI 172                 /SET PNTR TO TABLE AREA TO
LHI 000                 /STORE LIMIT OF TX BFR
LMB
JMP INCMD               /RETURN TO COMMAND MODE
/
```



"KILL" COMMAND SUBROUTINE FLOW CHART

THE "APPEND" COMMAND ROUTINE

THE "APPEND" COMMAND ROUTINE IS UTILIZED WHEN AN OPERATOR ENTERS THE COMMAND TO APPEND TEXT TO THE MAIN TEXT BUFFER. THIS ROUTINE FIRST CALLS UPON THE PREVIOUSLY DESCRIBED "CDIN" SUBROUTINE TO INPUT A LINE OF TEXT INTO THE TEMPORARY LINE TEXT BUFFER ON PAGE 00 (AS IT IS ENTERED VIA THE OPERATOR INPUT DEVICE). AFTER A LINE HAS BEEN STORED IN THE INPUT BUFFER, THE SUBROUTINE "CKOV" IS CALLED TO MAKE SURE THAT TRANSFERRING THE LINE INTO THE MAIN TEXT BUFFER WILL NOT CAUSE A MAIN BUFFER OVERFLOW CONDITION. PROVIDING THAT THERE IS NO OVERFLOW CONDITION, THE INFORMATION IN THE TEMPORARY LINE INPUT TEXT BUFFER IS "APPENDED" TO THE MAIN TEXT BUFFER STORAGE AREA. FOLLOWING THE TRANSFER, A ZERO BYTE IS WRITTEN TO SERVE AS AN END OF LINE TERMINATOR, THE TEXT BUFFER POINTER IS UPDATED TO SHOW THE NEW ADDRESS INDICATING THE END OF THE MAIN TEXT BUFFER, AND THE TEXT BUFFER LINE NUMBER (COUNTER) IS INCREMENTED. THIS TYPE OF PROCESS IS CONTINUED AS ADDITIONAL LINES OF TEXT ARE ENTERED. (EXITING FROM THE "APPEND" ROUTINE IS ACCOMPLISHED VIA THE "CDIN" INPUT ROUTINE WHEN A "CONTROL D" CHARACTER IS RECEIVED FROM THE OPERATOR'S INPUT DEVICE.)



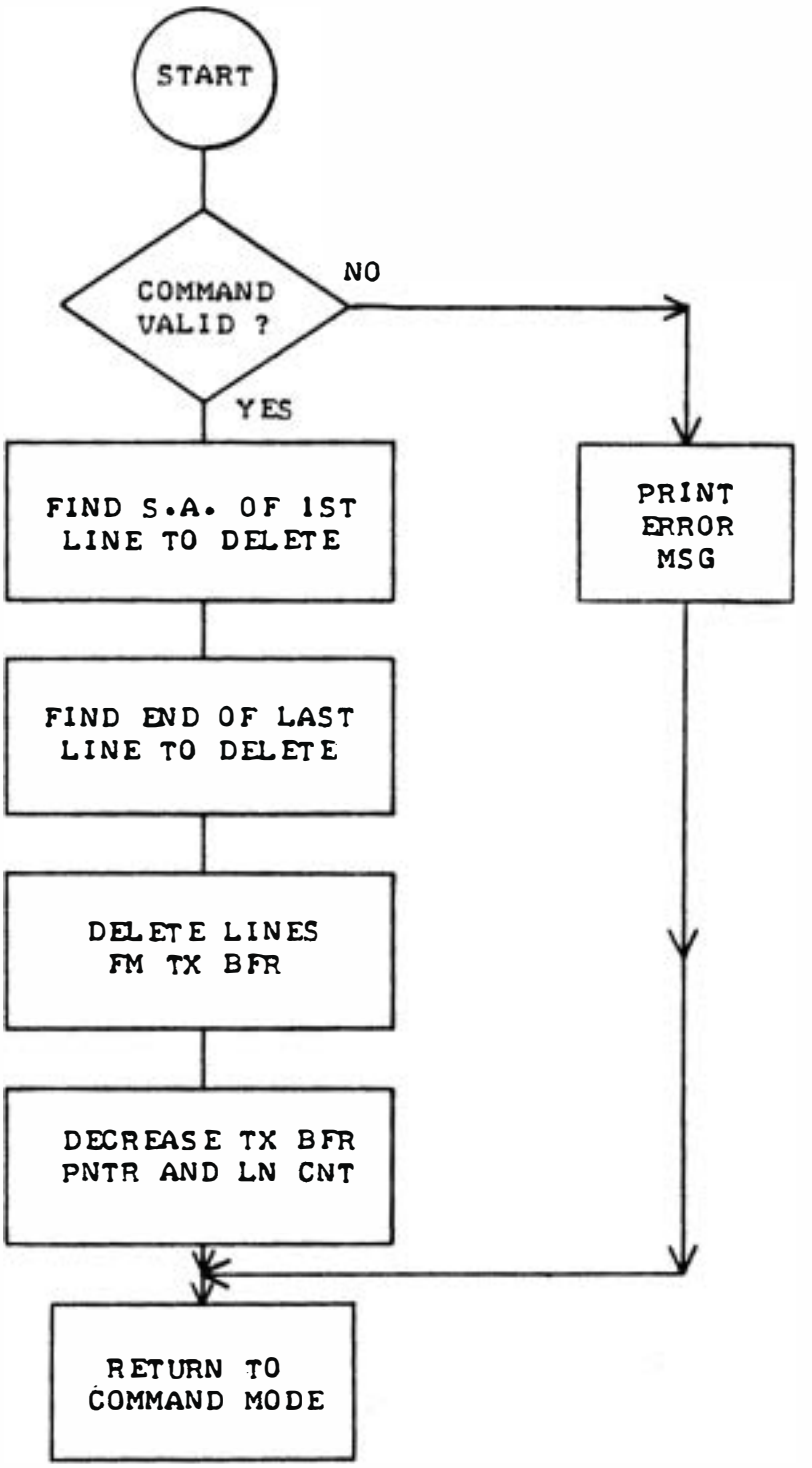
A BASIC FLOW CHART FOR THE "APPEND" ROUTINE IS SHOWN ON THE PRECEDING PAGE. A LISTING OF THE ROUTINE IS PRESENTED BELOW.

```
    MNEMONIC                COMMAND
 --------------          -------------

APND, CAL HDLN          /PRINT C/R,L/F
A1, CAL HDLN            /PRINT C/R, L/F
CAL CDIN                /INPUT LINE OF TEXT
CAL CKOV                /CHECK POSSIBLE BUFF OVRFLO
LDE                     /SAVE INP BFR PNTR
LAB
LCM                     /FETCH TEXT BFR PNTR
INL
LBM
LLA                     /SET UP S.A. OF INP BFR
A2, LHI 000
LAM                     /FETCH CHAR FROM INP BFR
LEL
LHB                     /SET UP TEXT BFR PNTR
LLC
LMA                     /STORE CHAR IN TEXT BFR
CAL INMEM               /INCR TEXT BFR PNTR
LBH                     /SAVE TEXT BFR PNTR
LCL
INE                     /INCR INP BFR PNTR
LAE                     /MOVE INP BFR PNTR TO ACC
CPD                     /LAST INP CHAR MOVED TO TX BFR?
JTZ A3                  /YES, WRAP UP LINE APPEND
LLE                     /NO, SET INP BFR PNTR
JMP A2                  /STORE MORE TEXT
A3, XRA                 /STORE THE END OF TEXT CHAR
LMA                     /IN THE TEXT BFR
INC                     /INCR TX BFR PNTR LO ADDR
JFZ A4                  /IF NOT 0, SKIP TO A4
INB                     /ELSE INCR PAGE PNTR
A4, LHA                 /SET PNTR TO TX BF PNTR STRAGE
LLI 160
LMC                     /SAVE NEW TX BFR PNTR LO ADDR
INL
LMB                     /AND PAGE ADDR
INL                     /SET PNTR TO LINE NUMBER
LAM                     /FETCH 1ST HALF OF LINE NUMBER
CAL INCR                /INCR LINE NUMBER
JMP A1                  /SET UP FOR NEW LINE TO APPEND
/
```

THE "DELETE" COMMAND ROUTINE

THE "DELETE" SUBROUTINE IS USED TO DELETE ONE OR MORE LINES OF TEXT FROM THE MAIN TEXT BUFFER. THIS IS ACCOMPLISHED BY ACTUALLY SHIFTING ALL THE LINES WHICH FOLLOW THE LINE(S) BEING DELETED "DOWNWARDS" IN THE MAIN TEXT BUFFER (FROM A HIGHER ADDRESS TO A LOWER ADDRESS) SO THAT THEY ARE WRITTEN OVER THE AREA FORMERLY OCCUPIED BY THE DELETED LINE(S). ALONG WITH THIS OPERATION, THE TEXT BUFFER POINTER IS CORRECTED TO INDICATE THE NEW ENDING LOCATION OF THE TEXT BUFFER, AND LINE NUMBER COUNTER IS DECREMENTED TO ACCOUNT FOR THE LINE(S) DELETED.

A FLOW CHART FOR THE "DELETE" SUBROUTINE IS SHOWN HERE:



THE "DELETE" ROUTINE WAS SET UP AS A SUBROUTINE FOR SEVERAL PURPOSES. FIRST, THE "CHANGE" ROUTINE USES THE "DELETE" SUBROUTINE TO REMOVE THE LINE(S) THAT ARE TO BE CHANGED FROM THE TEXT BUFFER. SECOND, THE SEARCH ROUTINE WILL ALSO CALL UPON A PORTION OF THE "DELETE" SUBROUTINE TO DELETE THE ORIGINAL LINE FROM THE TEXT BUFFER BEFORE INSERTING THE REVISED LINE.

THE LISTING FOR THE "DELETE" SUBROUTINE IS ILLUSTRATED ON THE NEXT PAGE.

```
   MNEMONIC                COMMENTS
-------------          -------------

DELET, CAL LDHILO     /SET LINE NUMBERS FROM CMND
CAL DLET              /DELETE LINES SPECIFIED
JMP INCMD             /RETURN TO COMMAND MODE
/
DLET, CAL FLO         /IS LO LIMIT = 0?
JTZ ERR               /YES, PRINT ERROR MSG
CAL FND               /FIND FIRST LINE TO DELETE
LCB                   /SAVE START ADDRESS
LBD
D3, CAL ZLOK          /FIND END OF LINE
CAL FLINO             /FETCH LINE NUMBER
CAL DECR              /DECR LINE NUMBER
LLI 165               /SET PNTR TO LINE COUNT
LAM                   /FETCH LINE COUNT
LLI 171               /SET PNTR TO HI LINE LIMIT
CPM                   /IS FINAL LINE TO DLT RCHED
JFZ D4                /NO, CONTINUE DELETE
DCL                   /IF 1ST HALF OF LN NO. MATCH
LAM                   /CHECK FOR MATCH OF OTHER HALF
LLI 164
CPM                   /IS FINAL LINE REACHED?
JTZ SDLT              /YES, COMPLETE DELETE MODE
D4, LLI 164           /NO, SET PNTR TO LINE COUNT
LAM
CAL INCR              /INCR LINE COUNT
JMP D3                /DELETE MORE LINES
SDLT, LLI 160         /SET PNTR TO START OF TBL AREA
LAM                   /FETCH CURRENT TX BFR PNTR
CPE                   /IS END OF TEXT BFR REACHED?
JFZ D6                /NO, SHIFT MORE TX DOWN IN BFR
INL                   /CHECK 2ND HALF OF PNTR'S
LAM                   /FOR A MATCH
CPD                   /DOES 2ND HALF MATCH?
JTZ D5                /YES,WRAP UP DELETE MODE
D6, CAL SHFT          /SHIFT CHAR DOWN IN TX BFR
CAL NCR               /INCR PNTR'S
LHI 000               /RESET TABLE PNTR
JMP SDLT              /CHECK FOR END OF SHIFT
D5, LMB               /SAVE NEW TX BFR PNTR
DCL                   /IN THE TABLE AREA
LMC
RET                   /RETURN TO CALLING PGM
/
```

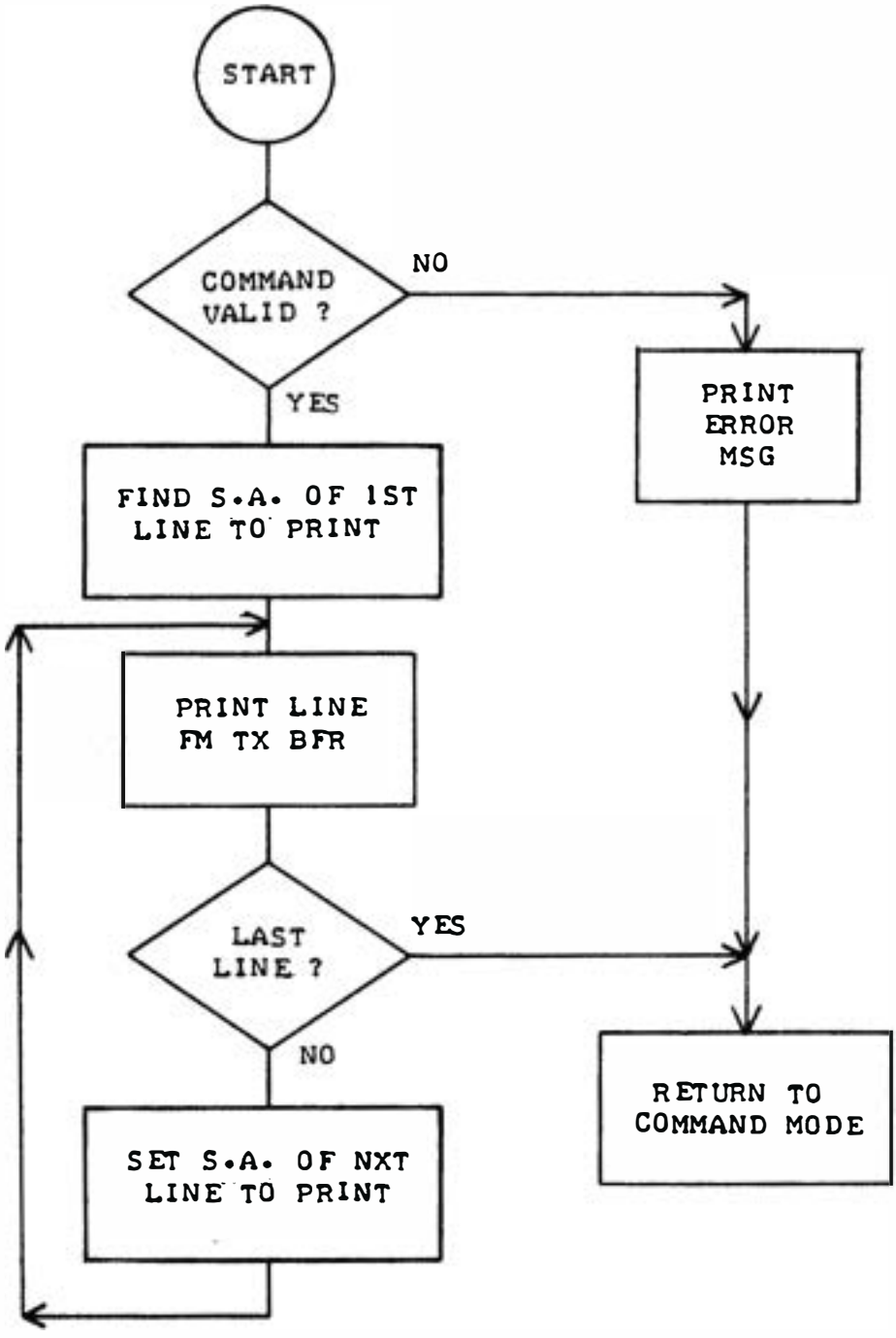
THE "LIST" COMMAND ROUTINE

THE "LIST" ROUTINE RESPONDS TO THE OPERATOR'S DIRECTIVE BY CAUSING THE SPECIFIED LINE(S) IN THE MAIN TEXT BUFFER TO BE DISPLAYED ON THE OUTPUT DEVICE. IF LINE(S) ARE SPECIFIED IN THE DIRECTIVE, THEN THE ROUTINE OPERATES BY SEARCHING THE MAIN TEXT BUFFER UNTIL IT REACHES THE FIRST LINE TO BE DISPLAYED. LINE(S) ARE THEN DISPLAYED BY USING THE PREVIOUSLY DESCRIBED UTILITY SUBROUTINE "MSG." THIS CONTINUES UNTIL THE LAST LINE SPECIFIED IN THE OPERATOR'S COMMAND HAS BEEN DISPLAYED.

IF NO LINE NUMBERS ARE DEFINED BY THE OPERATOR'S COMMAND, THE ROUT-

INE ASSUMES THAT THE ENTIRE CONTENTS OF THE MAIN TEXT BUFFER IS TO BE DISPLAYED.

A FLOW CHART FOR THE "LIST" ROUTINE IS SHOWN NEXT, FOLLOWED BY THE LISTING FOR THE ROUTINE.



MNEMONIC	COMMENTS
LIST, DCL	/DECR INP PNTR
LAL	
CPB	/LIST ALL OF TX BFR?
JTZ LALL	/YES, SET MAX LIMITS
CAL LDHILO	/NO, SET LINE NMBRS FROM CMND
CAL FLO	/IS LO LIMIT = 0?

```
MNEMONIC                COMMENTS
--------------          --------------

JTZ ERR                 /YES, PRINT ERROR MSG
JMP LST                 /START LIST ROUTINE
/
LALL, LLI 166           /SET PNTR TO LOW LIMIT STRAGE
LMI 001                 /SET START OF LIST TO LINE 1
INL
LMH
INL
LMH                     /SET HIGH LIMIT TO 000 000
INL                     /WHICH WILL BE CHANGED TO MAX
LMH                     /LINE NO. LATER
LST, CAL FLINO          /IS LINE NO. = 0?
JFZ LT1                 /NO, CONTINUE
INL                     /CHECK 2ND HALF OF LINE NO.
LAM
NDA                     /IS 2ND HALF = 0?
JTZ ERR                 /YES, PRINT ERROR MSG
LT1, LLI 170            /SET PNTR TO HI LIMIT
LAM                     /FETCH HI LINE LIMIT
NDA                     /HI LIMIT = 0?
JFZ LT2                 /NO, START LIST OUTPUT
INL                     /CHECK 2ND HALF OF HI LIMIT
LAM
NDA                     /2ND HALF = 0?
JFZ LT2                 /NO, START LIST OUTPUT
LLI 162                 /YES, SET PNTR TO LINE NO.
LAM                     /FETCH 1ST HALF OF LINE NO.
LLI 170                  /SET PNTR TO HI LIMIT
LMA                     /SET HI LIMIT TO LINE NO.
LLI 163
LAM                     /FETCH 2ND HALF OF LINE NO.
LLI 171                 /PNTR TO 2ND HALF OF HI LIMIT
LMA                     /SET HI LIMIT TO LINE NO.
LT2, CAL HDLN           /OUTPUT 2 C/R, L/F
CAL HDLN
CAL FND                 /FIND START ADDR OF LINE
LT3, LHD                /SET PNTR TO START OF LINE
LLE
CAL MSG                 /PRINT LINE OF TEXT
CAL INMEM               /INCR TEXT PNTR
LDH                     /SAVE TEXT PNTR
LEL
LHI 000                 /SET PNTR TO LO LIMIT
LLI 167
LAM                     /FETCH LO LIMIT
LLI 171                 /SET PNTR TO HI LIMIT
CPM                     /IS LO LIMIT = HI LIMIT?
JFZ LT4                 /NO, CONTINUE
DCL                     /CHECK 2ND HALF OF LO & HI LIM
LAM
LLI 166
CPM                     /IS LAST LINE LISTED?
JTZ INCMD               /YES, RET TO CMND MODE
LT4, LLI 166            /SET PNTR TO LO LIMIT
LAM                     /FETCH LO LIMIT
CAL INCR                /INCR LO LIMIT
```

```
   MNEMONIC                COMMENTS
 -------------           -------------

LLI 176                 /SET PNTR TO TMP STORAGE
LME                     /STORE TX PNTR IN TMP STORAGE
INL
LMD
CAL HDLN                /PRINT C/R, L/F
LLI 176                 /SET PNTR TO TMP STORAGE
LHI 000
LEM                     /FETCH TEXT BFR PNTR
INL
LDM
JMP LT3                 /CONTINUE PRINTING
/
```

THE "INSERT" COMMAND ROUTINE

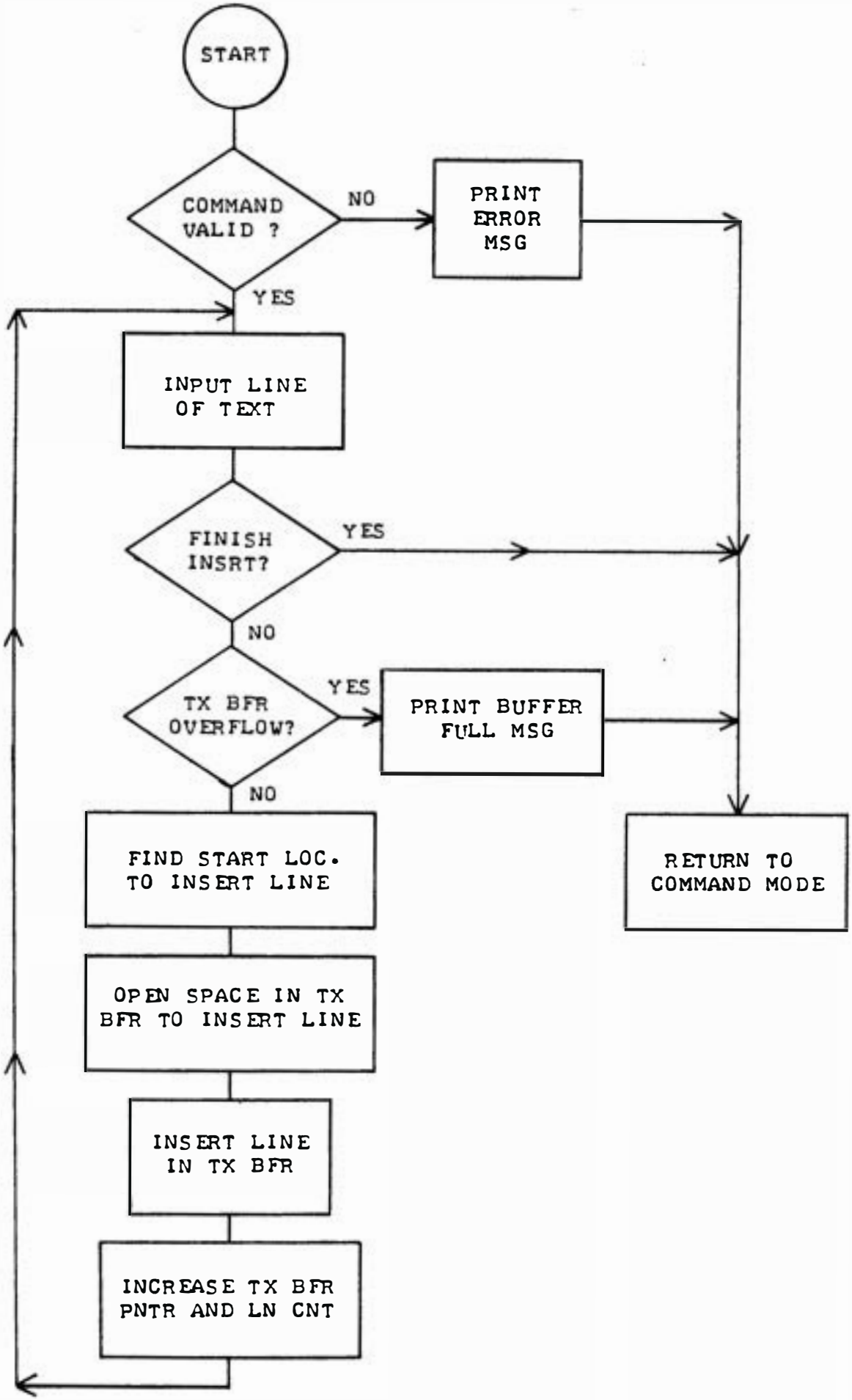
THE "INSERT" ROUTINE IS CALLED WHEN THE OPERATOR DESIRES TO HAVE THE EDITOR PROGRAM INSERT ONE OR MORE LINES IMMEDIATELY BEFORE A SPECIFIED LINE ALREADY EXISTING IN THE MAIN TEXT BUFFER. THE "INSERT" ROUTINE OPERATES IN THE FOLLOWING MANNER. FIRST, A LINE TO BE "INSERTED" IS READ INTO THE TEMPORARY INPUT BUFFER ON PAGE 00. THE LENGTH OF THE LINE IN THE TEMPORARY BUFFER IS THEN DETERMINED (AND PROVISION MADE FOR THE END OF LINE TERMINATING ZERO BYTE.) NEXT, "CKOV" IS CALLED TO SEE IF INSERTING THE NEW LINE WOULD CAUSE AN "OVERFLOW" CONDITION IN THE MAIN TEXT BUFFER. PROVIDING THAT NO "OVERFLOW" CONDITION WILL RESULT, ALL THE TEXT IN THE MAIN TEXT BUFFER, STARTING WITH THE LINE NUMBER SPECIFIED IN THE "INSERT" COMMAND DIRECTIVE, IS SHIFTED UPWARDS (TO HIGHER ADDRESSES) BY THE NUMBER OF LOCATIONS CONTAINED IN THE NEW LINE THAT IS TEMPORARILY RESIDING IN THE INPUT BUFFER. THEN, THE NEW LINE IS SIMPLY TRANSFERRED INTO THE "GAP" PROVIDED BY THE PREVIOUS TEXT SHIFTING OPERATION IN THE MAIN TEXT BUFFER. DURING THESE OPERATIONS, THE TEXT BUFFER POINTER IS UPDATED AND THE LINE NUMBER COUNTER IS ADVANCED TO TAKE ACCOUNT OF THE NEW LINE ENTRY.

AFTER ONE LINE HAS BEEN INSERTED, THE ROUTINE INCREMENTS THE NUMBER ORIGINALLY ENTERED BY THE OPERATOR AS THE "INSERT BEFORE" LINE NUMBER AND THE ABOVE OPERATION MAY BE REPEATED. THUS, ONE MAY INSERT SEVERAL LINES AT A GIVEN POINT BY SIMPLY ENTERING THE LINES ON THE OPERATOR'S INPUT DEVICE.

AS IN THE "DELETE" ROUTINE DESCRIBED EARLIER, THE "INSERT" ROUTINE HAS BEEN SET UP AS A SUBROUTINE. IT IS CALLED UPON BY THE "CHANGE" COMMAND ROUTINE AS WELL AS THE "SEARCH" ROUTINE IN ADDITION TO IT'S "STAND ALONE" CAPABILITY AS DESCRIBED.

THE DETAILED LISTING OF THE "INSERT" SUBROUTINE AND A FLOW CHART OF THE ROUTINE'S OPERATION IS PRESENTED ON THE NEXT SEVERAL PAGES.

```
   MNEMONIC                COMMAND
 -------------           -------------

INSRT, CAL LDHILO       /SET LINE NUMBERS FROM CMND
INS1, CAL NSRT          /INSERT LINE INTO TEXT BFR
JMP INS1                /SET UP FOR ANOTHER INSERT
/
```



"INSERT" SUBROUTINE FLOW CHART

```
MNEMONIC                COMMENTS
--------------          --------------

NSRT, CAL FLO
JTZ ERR                 /YES, PRINT ERROR MSG
CAL HDLN                /PRINT C/R, L/F
CAL CDIN                /INPUT NEW LINE TO INSERT
CAL CKOV                /CHECK FOR POSSIBLE BFR OVRFLO
CAL FND                 /FIND START OF LINE
LLI 176                 /SET PNTR TO TMP STORAGE
LME                     /SAVE START ADDR OF LINE
INL
LMD
SNST, LLI 160           /SET PNTR TO TX BF PNTR
LAM                     /FETCH TX BFR PNTR
INL                     /SET REG D & E AS 'FROM PNTR'
LDM
LEA                     /SET UP END OF LINE PNTR
ADC                     /BY ADDING CHAR COUNT + 1
LCA                     /SET REG B & C AS 'TO PNTR'
LBD
JFC NS1                 /INCR 'TO' PAGE PNTR? NO
INB                     /YES
LMB                     /SET NEW TX BFR PNTR
NS1, DCL
LMC
CAL CKTX                /'FROM' = END OF TX BFR?
JTZ NST1                /YES, INSERT LINE
NS2, LAE                /NO, DECR 'FROM'
SUI 001
LEA                     /DOES 2ND HALF NEED DECR
JFC NS3                 /NO, SKIP DECR
DCD                     /YES, DECR 2ND HALF OF 'FROM'
NS3, LAC                /DECR 'TO'
SUI 001
LCA                     /DOES 2ND HALF NEED DECR
JFC NS4                 /NO, SKIP DECR
DCB                     /YES, DECR 2ND HALF
NS4, CAL CKTX           /'FROM' = START OF INSERT LOC?
JTZ NSTL                /YES, INSERT LINE
CAL SHFT                /NO, SHFT CHAR IN TX BFR
JMP NS2                 /CONTINUE SHIFTING
/
NSTL, CAL SHFT          /SHFT CHAR BEFORE NEW LINE
NST1, LLI 173           /SET PNTR TO CHAR COUNT+1
LHI 000
LAM                     /FETCH CHAR COUNT+1
SUI 001                 /DECR TO GET CHAR COUNT
LLI 176                 /SET PNTR TO TMP STORAGE
ADM                     /ADD S.A. TO CHAR COUNT
LMA                     /SAVE FINAL ADDR
LCE                     /SET 'TO PNTR'
LAD
CAL FBFLM               /FETCH START OF INP BFR
LDH                     /SET 'FROM PNTR'
LEB
LBA                     /SAVE FINAL PAGE IN REG B
NS6, CAL SHFT           /SHIFT INP TO TX BFR
CAL NCR                 /INCR 'TO' AND 'FROM'
LLI 176                 /SET PNTR TO TMP STORAGE
LHI 000
```

```
    MNEMONIC                COMMENTS
 -------------          --------------

LAM                     /FETCH FINAL LOC IN TX BFR
CPC                     /END OF LINE INPUT?
JFZ NS6                 /NO, CONT. STORAGE
LLC                     /YES, SET END OF LINE PNTR
LHB
LMI 000                 /STORE EOL INDICATOR
LLI 166                 /SET PNTR TO LO LIMIT
LHI 000
LAM                     /FETCH LO LIMIT
CAL INCR                /INCR LO LIMIT
LLI 162                 /SET PNTR TO LINE NO.
LAM                     /FETCH LINE NO.
CAL INCR                /INCR LINE NO.
RET                     /RETURN TO CALLING PGM
/
CKTX, LLI 176           /SET PNTR TO TMP STORAGE
LHI 000
LAE
CPM                     /TMP STORAGE = 'FROM'?
RFZ                     /NO,RET WITH Z FLAG RESET
INL
LAD                     /CHECK 2ND HALF OF ADDR
CPM
RET                     /IF =,RET WITH Z FLAG SET
/
```

THE "CHANGE" COMMAND ROUTINE

THE READER WHO HAS BEEN DILIGENTLY STUDYING EACH ROUTINE AS THEY HAVE BEEN PRESENTED, AND POSSIBLY THOUGHT THAT FROM TIME TO TIME, THE ROUTINES SEEMED TO BE A LITTLE COMPLICATED OR HARD TO FOLLOW, WILL UNDOUBTABLY BE DELIGHTED TO OBSERVE THE FOLLOWING "CHANGE" ROUTINE. THE ROUTINE IS USED TO "CHANGE" ONE OR MORE LINES SPECIFIED IN THE COMMAND. HOWEVER, IT CONSIST OF NOTHING MORE THAN A CALL TO THE "DELETE" SUBROUTINE TO EFFECTIVELY "REMOVE" THE ORIGINAL LINES BEING CHANGED FROM THE TEXT BUFFER, FOLLOWED BY A JUMP TO THE "INSERT" SUBROUTINE JUST DESCRIBED IN THE PREVIOUS SECTION! (THIS LITTLE ROUTINE DEMONSTRATES HOW COMPUTER PROGRAMS CAN RAPIDLY INCREASE IN CAPABILITY ONCE A "BASE" OF OTHER SUBROUTINES HAS BEEN ESTABLISHED.) THE THREE INSTRUCTIONS THAT MAKE UP THE "CHANGE" ROUTINE ARE SHOWN BELOW. HOWEVER, TO CONSERVE A LITTLE SPACE, AND TO AVOID BORING THE READER, A FLOW CHART HAS NOT BEEN INCLUDED WITH THE LISTING!

```
    MNEMONIC                COMMENTS
 -------------          --------------

/
CHANGE, CAL LDHILO      /SET LINE NUMBERS FROM CMND
CAL DLET                /DELETE LINES SPECIFIED
JMP INS1                /INSERT NEW LINES
/
```

THE "SEARCH" COMMAND ROUTINE

ACTUALLY, THE "CHANGE" ROUTINE JUST PRESENTED WAS STRATEGICALLY LOCATED IN THIS PRESENTATION TO GIVE THE READER A "BREATHER" BEFORE GETTING INTO WHAT IS PROBABLY THE MOST COMPLEX ROUTINE IN THE EDITOR PROGRAM - THE "SEARCH" ROUTINE. (IF YOUR NOT READY FOR THIS, TAKE A FEW MORE MINUTES ENJOYING THE "NICE-N-EASY" CHANGE ROUTINE!) ACTUALLY, THOUGH, THE SEARCH ROUTINE, WITH THE BACKGROUND THE READER NOW HAS, SHOULD NOT BE ALL THAT DIFFICULT TO COMPREHEND.

THE "SEARCH" ROUTINE STARTS OUT BY EXAMINING THE CONTENTS OF THE LINE SPECIFIED IN THE COMMAND SEQUENCE, LOOKING FOR THE FIRST APPEARANCE OF THE "SEARCH CHARACTER" DEFINED BY THE OPERATOR. AS THE LINE IS BEING SEARCHED, EACH CHARACTER IN THE LINE IS ALSO TRANSMITTED TO THE DISPLAY DEVICE FOR REVIEW BY THE OPERATOR. AND, AS THIS OCCURS, THE CHARACTERS ARE ALSO STORED IN THE INPUT TEXT BUFFER AREA.

WHEN THE FIRST APPEARANCE OF THE "SEARCH CHARACTER" IS DETECTED, THE CURRENT CONTENTS OF CPU REGISTERS "B" THROUGH "E" ARE SAVED FOR LATER USE IN CONNECTION WITH THE SEARCH ROUTINE IN A TEMPORARY STORAGE AREA ON PAGE 00. THEN, THE STANDARD TEXT INPUT BUFFER SUBROUTINE IS CALLED TO ALLOW THE OPERATOR TO INPUT ANY CORRECTIONS TO THE LINE.

AT THIS POINT, THERE ARE SEVERAL OPTIONS AVAILABLE TO THE OPERATOR. FIRST, THE REMAINDER OF THE LINE MAY BE REVISED BY THE OPERATOR SIMPLY ENTERING THE REVISED TEXT ON THE OPERATOR'S INPUT DEVICE AND TERMINATING THE NEW ENTRY WITH A CARRIAGE RETURN. THIS OPTION ESSENTIALLY UTILIZES PART OF THE "DELETE" SUBROUTINE TO REMOVE THE ORIGINAL LINE FROM THE MAIN TEXT BUFFER AND THEN PART OF THE "INSERT" SUBROUTINE TO PLACE THE NEW MODIFIED LINE (FROM THE TEXT INPUT BUFFER) BACK INTO THE MAIN TEXT BUFFER.

OR, THE OPERATOR MAY ELECT SEVERAL DIFFERENT TYPES OF FUNCTIONS BY USING SPECIAL "CONTROL CHARACTER" COMMANDS. FOR INSTANCE, ONE SUCH OPTION, THE CAPABILITY OF HAVING THE SEARCH ROUTINE PROCEED TO THE NEXT OCCURENCE OF THE "SEARCH CHARACTER" IN THE LINE, IS INITIATED BY THE OPERATOR ENTERING A "CONTROL T" FOLLOWED BY A "CONTROL L" ON THE OPERATOR INPUT DEVICE. WHEN THIS IS DONE, THE "SEARCH" ROUTINE PROCEEDS AS FOLLOWS.

WHEN THE "CONTROL T" CHARACTER IS RECEIVED BY THE INPUT TEXT BUFFER ROUTINE IT IS STORED AS A "TEXT" CHARACTER (BECAUSE IT IS NOT ONE OF THE SPECIAL "CONTROL" CHARACTERS PREVIOUSLY DEFINED IN THE DISCUSSION ON THAT SUBROUTINE). HOWEVER, RECEIPT OF THE "CONTROL L" CHARACTER BY THE INPUT ROUTINE CAUSES THE SUBROUTINE TO BE EXITED AS IT SERVES AS AN "END OF INPUT" TERMINATOR. WHEN PROGRAM OPERATION RETURNS TO THE "SEARCH" ROUTINE (WHICH CALLED THE TEXT INPUT ROUTINE), THE SEARCH ROUTINE CHECKS THE LAST CHARACTER ENTERED IN THE INPUT TEXT BUFFER TO DETERMINE IF IT IS ONE OF SEVERAL "CONTROL CHARACTERS" AS DEFINED IN THE PRESENT DISCUSSION. ONE SUCH CONTROL CHARACTER THAT IT CHECKS FOR IS THE "CONTROL T" (ASCII CODE 224) SYMBOL. IF THE "CONTROL T" IS DETECTED, THE SEARCH ROUTINE RESTORES THE PREVIOUSLY SAVED CPU REGISTERS "B" THROUGH "E" BACK TO THE VALUES THEY CONTAINED WHEN THE SEARCH CHARACTER WAS ORIGINALLY FOUND. THE ROUTINE THEN PROCEEDS TO CONTINUE TO SEARCH THE REMAINDER OF THE ORIGINAL LINE FOR THE NEXT OCCURENCE OF THE SEARCH CHARACTER. (AS THIS IS DONE, CHARACTERS ARE AGAIN DISPLAYED, AND STORED IN THE TEMPORARY TEXT INPUT BUFFER, WITH THE FIRST SUCH CHARACTER BEING PLACED WHERE THE CODE FOR THE "CONTROL T" WAS IN THE TEXT INPUT BUFFER - THUS EFFECTIVELY REMOVING THE "CONTROL T" FROM THE INPUT BUFFER!)

ANOTHER OPTION A USER HAS IS TO ENTER A "CONTROL N" (ASCII CODE 216) INSTEAD OF "CONTROL T," FOLLOWED BY A "CONTROL L." THIS SEQUENCE IS INTERPRETED IN A SIMILAR FASHION AS DESCRIBED FOR THE "CONTROL T" OPTION EXCEPT THAT NOW THE SEARCH ROUTINE WILL FIRST CALL THE OPERATOR DEVICE INPUT ROUTINE TO OBTAIN A "NEW" SEARCH CHARACTER. THIS NEW CHARACTER IS THEN USED TO CONTINUE SEARCHING THE REMAINDER OF THE ORIGINAL LINE IN THE SAME FASHION AS THAT JUST DESCRIBED FOR CONTINUEING THE SEARCH FOR ANOTHER OCCURENCE OF THE SAME SEARCH CHARACTER.

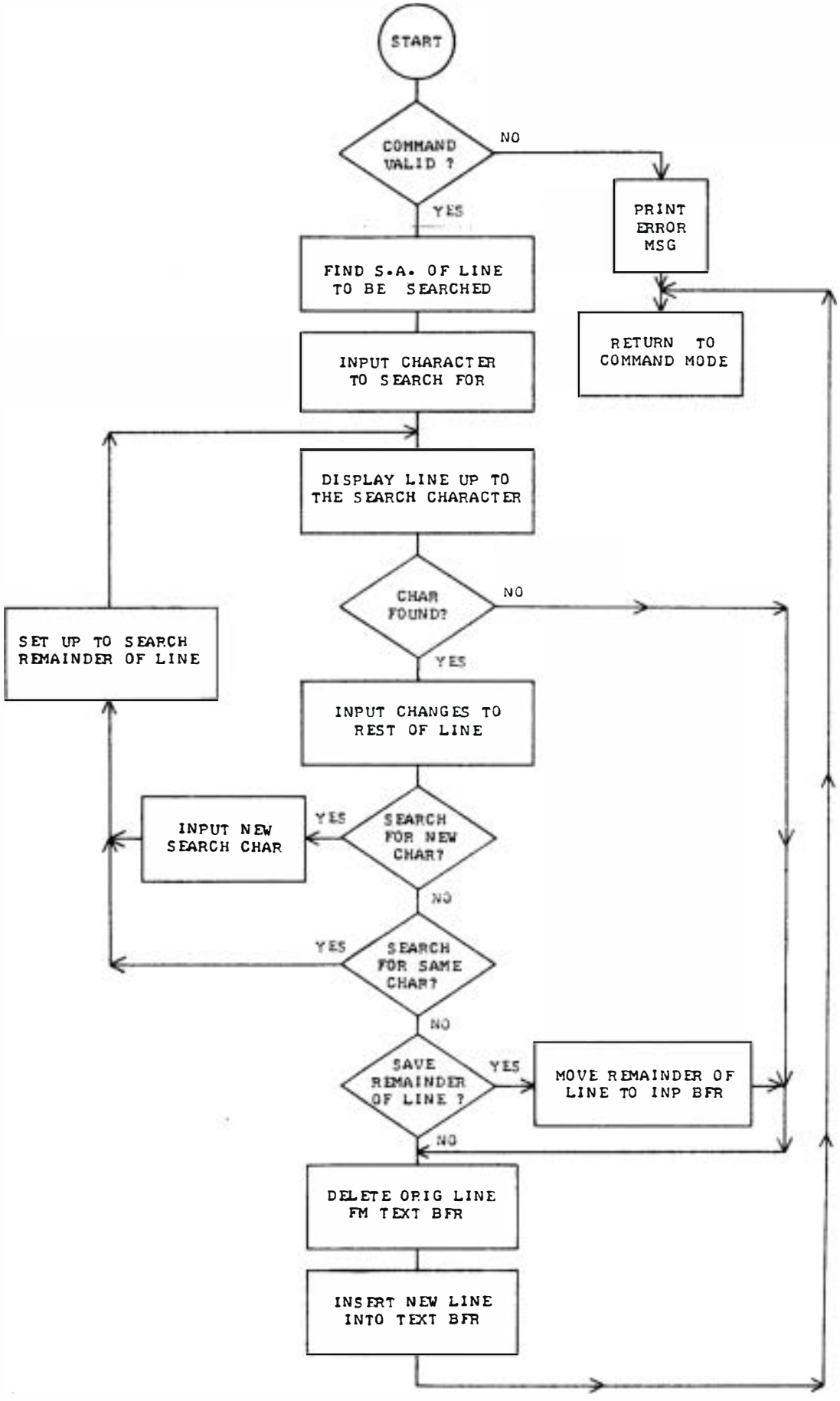
STILL ANOTHER OPTION AVAILABLE IS FOR THE USER TO ENTER A "CONTROL U" (ASCII CODE 225) FOLLOWED BY A "CONTROL L." FOR THIS CASE, THE ROUTINE WILL PROCEED TO MOVE THE REMAINDER OF THE ORIGINAL LINE FROM THE MAIN BUFFER INTO THE INPUT BUFFER! (NOTE THAT CONSIDERABLE MANIPULATING POWER IS BEING PROVIDED HERE. THE "CONTROL U" FUNCTION MAY BE USED SHOULD AN OPERATOR DECIDE NOT TO CHANGE THE ORIGINAL CONTENTS OF A LINE AFTER HAVING REACHED A SEARCH CHARACTER, OR, THE OPERATOR MAY HAVE MADE CHANGES IN THE MIDDLE OF A LINE. (ADDITIONS OR DELETIONS - REMEMBER, ALL THE CAPABILITIES OF THE "INPUT" SUBROUTINE CAN STILL BE USED, SUCH AS THE "RUBOUT" CHARACTER TO DELETE CHARACTERS IN THE LINE THAT PRECEED THE "SEARCH CHARACTER.") THIS NESTING OF ONE POWERFUL SUBROUTINE WITHIN ANOTHER TENDS TO MAKE EXPLANATION OF THE "SEARCH" ROUTINE SOMEWHAT COMPLICATED.)

ONCE A REVISED LINE IS IN IT'S FINAL FORM IN THE INPUT TEXT BUFFER AND A LINE TERMINATING CHARACTER DETECTED (WHICH IS NOT PRECEEDED BY A "CONTROL T, OR AN N") THE SEARCH ROUTINE PREPARES TO ENTER THE REVISED LINE INTO THE MAIN TEXT BUFFER. A PORTION OF THE PREVIOUSLY DISCUSSED "DELETE" ROUTINE IS CALLED UPON TO REMOVE THE OLD ORIGINAL LINE FROM THE MAIN TEXT BUFFER. THEN THE "INSERT" ROUTINE IS CALLED UPON TO INSERT THE NEW INFORMATION INTO THE MAIN TEXT BUFFER. THE TEXT BUFFER POINTER IS REVISED AS REQUIRED TO COMPENSATE FOR THE OPERATIONS. ADDITIONALLY, THE "SEARCH" ROUTINE DECREMENTS THE LINE NUMBER. THIS IS DONE BECAUSE THE "INSERT" SUBROUTINE CALLED DECREMENTS THE LINE NUMBER WHEN THE NEWLY REVISED LINE IS ADDED TO THE TEXT BUFFER. THUS, THE LINE NUMBER REMAINS THE SAME WHEN A "SEARCH" COMMAND IS USED TO REVISE A LINE.

A FEW SPECIAL POINTS ABOUT THE SEARCH ROUTINE SHOULD BE MADE. THE READER SHOULD UNDERSTAND THAT IF THE SEARCH ROUTINE SHOULD LOCATE THE END OF LINE TERMINATOR (ZERO BYTE) IN THE MAIN TEXT BUFFER WHILE SCANNING FOR A "SEARCH CHARACTER" THE SEARCH WILL NATURALLY BE HALTED. THE ROUTINE WILL THEN PROCEED TO DELETE THE ORIGINAL LINE IN THE MAIN TEXT BUFFER AND THEN INSERT THE CONTENTS OF THE INPUT TEXT BUFFER INTO THE MAIN TEXT BUFFER. THE REASON FOR FOLLOWING THIS PROCEDURE, WHICH MIGHT SEEM EXTRANEOUS AT FIRST GLANCE, IS SO THAT IF CHANGES HAVE BEEN MADE IN THE LINE, THE REVISIONS WILL BE STORED IN THE MAIN TEXT BUFFER. (REMEMBER, ONE COULD HAVE A SITUATION WHERE A PARTICULAR SEARCH CHARACTER WAS SPECIFIED AND FOUND. THEN CHANGES WERE MADE. THEN A SECOND SEARCH CHARACTER WAS SPECIFIED AND NOT FOUND.)

SECONDLY, THERE ARE TWO TIMES IN THE DESCRIBED SEARCH ROUTINE WHERE THE USER PROVIDED "RCV" SUBROUTINE IS USED DIRECTLY. SPECIFICATIONS FOR THIS SUBROUTINE DICTATED THAT IT ECHO THE INPUT (FOR GENERAL USE IN THE PROGRAM). THIS ECHO CAPABILITY DOES NO HARM IN THE SEARCH ROUTINE, HOWEVER SOME USERS MAY FIND IT ANNOYING TO HAVE THE SEARCH CHARACTER ECHO BACK WHEN THEY CHANGE THE SEARCH CHARACTER IN THE MIDDLE OF A LINE. THOSE WHO DESIRE TO DO SO, MIGHT UTILIZE A SPECIAL INPUT SUBROUTINE RATHER THAN THE "RCV" SUBROUTINE INDICATED IN ORDER TO DISABLE THE ECHO FUNCTION DURING THE INPUTTING OF A "SEARCH CHARACTER" TO THE SEARCH ROUTINE.

A FLOW CHART OF THE "SEARCH" ROUTINE IS SHOWN ON THE NEXT PAGE.


START
COMMAND VALID ?
NO
YES
PRINT ERROR MSG
RETURN TO COMMAND MODE
FIND S.A. OF LINE TO BE SEARCHED
INPUT CHARACTER TO SEARCH FOR
DISPLAY LINE UP TO THE SEARCH CHARACTER
CHAR FOUND?
NO
YES
SET UP TO SEARCH REMAINDER OF LINE
INPUT CHANGES TO REST OF LINE
INPUT NEW SEARCH CHAR
YES
SEARCH FOR NEW CHAR?
NO
YES
SEARCH FOR SAME CHAR?
NO
SAVE REMAINDER OF LINE ?
YES
MOVE REMAINDER OF LINE TO INP BFR
NO
DELETE ORIG LINE FM TEXT BFR
INSERT NEW LINE INTO TEXT BFR

A LISTING OF THE "SEARCH" ROUTINE IS SHOWN BELOW.

```
    MNEMONIC                COMMENTS
 --------------          --------------

SRCH, CAL LDHILO     /SET LINE NUMBERS FROM CMND
CAL FLO              /IS LO LIMIT = 0?
JTZ ERR              /YES, PRINT ERROR MSG
CAL FND              /FIND S.A. OF LINE
LCE                  /SAVE START ADDR
LBD
LLI 176              /SET PNTR TO TMP STORAGE
LHI 000
LME                  /SAVE S.A. IN TMP STORAGE
INL
LMD
LLE                  /SAVE S.A. IN REG'S H & L
LHD
CAL RCV              /INP CHAR TO SEARCH FOR
LDH                  /RESTORE S.A.
CAL FBFLM            /FETCH S.A. OF INP BFR
LCB                  /SAVE S.A. OF INP BFR
LBA                  /SAVE SEARCH CHAR
LLI 144              /SAVE REG'S IN SEARCH TBL
CAL SVBE
CAL HDLN             /PRINT C/R,L/F
LLI 144
CAL LDBE             /RESTORE REG'S
S1, LLE              /SET PNTR TO TEXT LINE
LHD
LAM                  /FETCH CHAR FROM TEXT
NDA                  /CHAR = EOL INDICATOR?
JTZ SCT              /YES, STORE LINE IN TX BFR
LLC                  /SET PNTR TO INP BFR
LHI 000
LMA                  /STORE CHAR IN INP BFR
LLI 144
CAL SVBE             /SAVE REG'S
CAL PRINT            /PRINT CHAR
LLI 144
CAL LDBE             /RESTORE REG'S
CAL NCRD             /INCR TEXT PNTR
INC                  /INCR INP BFR PNTR
JTZ SCT              /=0? YES, STORE LINE IN TX BFR
CPB                  /NO, IS CHAR = SRCH CHAR?
JFZ S1               /NO, CONT. SRCHING LINE
LLC                  /SET PNTR TO INP BFR
CAL IN2              /INP CHANGES FROM KYBD
DCE                  /BACK UP INP BFR PNTR
LLE                  /SET PNTR TO LAST CHAR INP
LAM                  /FETCH LAST CHAR INP
CPI 216              /CNT'L N?
JTZ NC               /YES, SRCH FOR NEW CHAR
CPI 224              /CNT'L T?
JTZ NX               /YES, SRCH FOR SAME CHAR
CPI 225              /CNT'L U?
JTZ SV               /YES, SAVE REST OF ORGIN. LINE
INE                  /RESET INP BFR PNTR
```

```
    MNEMONIC              COMMENTS
 -------------          -------------

S3, CAL CKOV          /CHECK FOR POSSIBLE BFR OVRFLO
CAL FND               /FETCH S.A. OF SRCH LINE
LBD                   /SAVE S.A. IN REG B & C
LCE
CAL ZLOK              /FETCH S.A. OF NEXT LINE
LHI 000               /SET PG TO 000
CAL SDLT              /DELETE ORIGINAL LINE
LLI 173
LCM                   /FETCH CHAR COUNT+1
CAL SNST              /INSERT LINE
LLI 162               /SET PNTR TO LINE NO.
LAM
CAL DECR              /DECR LINE NO. TO CORRECT FOR
/                      INCR IN SNST ROUTINE
JMP INCMD             /RET TO COMMAND MODE
/
NC, CAL RCV           /FETCH NEW SRCH CHAR
LEL                   /SAVE INP BFR PNTR
LLI 144               /SAVE IN SRCH TBL
LMA
/
NX, CAL RT            /RESET REG FROM SRCH TBL
JMP S1                /CONT. SRCH
/
RT, LAE               /SAVE INP BFR PNTR
LLI 144
CAL LDBE              /LOAD REG FROM SRCH TBL
LCA                   /SET INP BFR PNTR
JMP NCRD              /INCR TX BFR PNTR AND RET
/
SV, CAL RT            /ERSET REG FROM SRCH TBL
S4, LLE               /SET TX BFR PNTR
LHD
LAM                   /FETCH NEXT CHAR FROM TX BFR
NDA                   /IS CHAR = EOL INDICATOR?
JTZ SCT               /YES, COMPLETE SRCH
INC
DCC                   /IS INP BFR FULL?
JTZ S5                /YES, IGNORE CHAR
LLC                   /NO, SET INP BFR PNTR
LHI 000
LMA                   /STORE CHAR IN INP BFR
INC                   /INCR INP BFR PNTR
S5, CAL NCRD          /INCR TX BFR PNTR
JMP S4                /CONT. STORING LINE
/
SCT, LLC              /SAVE INP BFR PNTR
CAL FBFLM             /FETCH INP BFR LIMIT
JMP S3                /INSERT NEW LINE IN TXT BFR
```

THE "WRITE" COMMAND ROUTINE

THE "WRITE" ROUTINE INCLUDED IN THE EDITOR PROGRAM SIMPLY PROVIDES A "SET UP" FUNCTION AND THEN CALLS ON THE USER PROVIDED BULK STORAGE OUTPUT ROUTINE SO THAT THE CONTENTS OF THE MAIN TEXT BUFFER MAY BE SAVED

FOR LATER RECALL. AS THE READER CAN SEE BY EXAMINING THE LISTING FOR THE ROUTINE ILLUSTRATED BELOW, THE ROUTINE SIMPLY TRANSFERS THE CURRENT TEXT BUFFER POINTER (INDICATING THE LAST LOCATION USED BY THE TEXT BUFFER) AND THE CURRENT LINE NUMBER FROM PAGE 00 TO THE FIRST FOUR LOCATIONS ON PAGE 07. AS MENTIONED EARLIER, THOSE LOCATIONS ARE CONSIDERED AS PART OF THE MAIN TEXT BUFFER BUT ARE RESERVED SPECIFICALLY FOR THIS PURPOSE. THE ROUTINE THEN SETS UP CPU REGISTERS "H" & "L" TO THE STARTING ADDRESS OF THE MAIN TEXT BUFFER, AND CPU REGISTERS "D" & "E" TO THE ENDING ADDRESS (CURRENT TEXT BUFFER POINTER). THIS IS DONE SO THAT THE BULK STORAGE WRITE ROUTINE (PROVIDED BY THE USER) HAS THIS INFORMATION READILY AVAILABLE. THE ROUTINE THEN CALLS THE BULK STORAGE WRITE ROUTINE WHICH IT EXPECTS TO HANDLE THE PROCESS OF WRITING THE COMPLETE CONTENTS OF THE MAIN TEXT BUFFER ON THE EXTERNAL BULK STORAGE DEVICE. WHEN THE BULK STORAGE OUTPUT ROUTINE IS FINISHED (IT SHOULD BE IN THE FORM OF A SUBROUTINE) THE WRITE ROUTINE SHOWN SIMPLY RETURNS CONTROL OF THE PROGRAM TO THE EDITOR COMMAND MODE. THE SHORT LISTING FOR THE ROUTINE IS SHOWN HERE:

```
    MNEMONIC                COMMENTS
 -------------          --------------

WRITE, LLI 160        /SET PNTR TO TX BF PNTR & LINE NO.
LHI 000
CAL LDBE              /FETCH TX BF PNTR & LINE NO.
LLH                   /SET PNTR TO START OF TEXT BFR AREA
LHI 007
CAL SVBE              /STORE TEXT BF PNTR & LINE NO.
LLI 000               /TO BE WRITTEN TO BULK STORAGE
LDC                   /SET START AND END ADDR FOR
LEB                   /LIMITS OF TX BFR TO BE STORED
CAL PUNCH             /WRITE TX BFR TO BULK STORAGE
JMP INCMD             /RETURN TO COMMAND MODE
/
```

NOTES AND SUGGESTIONS FOR THE USER PROVIDED BULK STORAGE OUTPUT ROUTINE

THE ACTUAL ROUTINE THAT OUTPUTS THE CONTENTS TO THE BULK STORAGE DEVICE WILL, NATURALLY, BE A FUNCTION OF THE TYPE OF BULK STORAGE DEVICE BEING USED. SUCH A DEVICE COULD RANGE FROM A SIMPLE PAPER TAPE PUNCH SYSTEM, TO A MAGNETIC TAPE STORAGE UNIT, OR EVEN POSSIBLY A DISC UNIT. REGARDLESS OF WHAT TYPE OF UNIT IS TO BE USED, SOME CONSIDERATION MUST BE GIVEN TO THE INTENDED LONG RANGE USE OF THE TEXT MATERIAL THAT IS TO BE STORED ON THE BULK STORAGE DEVICE. IF THE BULK STORAGE CAPABILITY IS TO BE USED ONLY FOR THE PURPOSE OF BEING ABLE TO RELOAD THE TEXT BACK INTO THE EDITOR PROGRAM FOR LATER REVISIONS, THEN THE ACTUAL WRITE PROCESS CAN BE QUITE SIMPLE AND STRAIGHT FORWARD. ONE NEEDS MERELY "DUMP" A VIRTUAL IMAGE OF THE MAIN TEXT BUFFER TO THE STORAGE DEVICE AND HAVE THE BULK STORAGE INPUT ROUTINE PERFORM THE REVERSE.

HOWEVER, IN CERTAIN CASES, TEXT PREPARED USING THE EDITOR MAY BE INTENDED FOR LATER USE WITH OTHER PROGRAMS OR IN OTHER "SYSTEMS" (SUCH AS, SAY, THE INPUT TO A NUMERICALLY CONTROLLED MACHINE TOOL). IN SUCH CASES, THERE ARE A FEW ITEMS THAT THE PREPARER OF THE BULK STORAGE OUTPUT ROUTINE SHOULD GIVE SOME THOUGHT TO.

FOR INSTANCE, ONE MIGHT DESIRE TO PREPARE "SOURCE LISTINGS" USING THE EDITOR PROGRAM AND THEN HAVE THESE SOURCE LISTINGS PROCESSED BY

ANOTHER TYPE OF PROGRAM REFERRED TO AS AN "ASSEMBLER" PROGRAM. SUPPOSE, FOR EXAMPLE, THAT ONE DESIRED TO PREPARE PUNCHED PAPER TAPE OF THE TEXT IN THE MAIN TEXT BUFFER, AND LATER HAVE THAT TAPE READ BY AN INPUT ROUTINE TO THE "ASSEMBLER" PROGRAM. IT MIGHT NOT BE APPROPRIATE TO SIMPLY PERFORM A VIRTUAL DUMP OF THE CONTENTS OF THE EDITOR TEXT BUFFER BECAUSE OF THE FOLLOWING FACTOR. THE "END OF LINE TERMINATOR" IN THE MAIN TEXT BUFFER IS SIMPLY A "ZERO BYTE." THE "ASSEMBLER" PROGRAM MIGHT REQUIRE THE END OF LINE TERMINATOR TO BE THE CODE FOR A CARRIAGE RETURN - OR EVEN A COMBINATION OF A CARRIAGE RETURN AND A LINE FEED! THUS, IF ONE INTENDED TO USE THE MATERIAL OUTPUTTED BY THE BULK STORAGE ROUTINE FOR SUCH A PURPOSE, ONE WOULD WANT TO HAVE THE OUTPUT ROUTINE SCAN THE TEXT AS IT WAS BEING PROCESSED AND MAKE AN APPROPRIATE SUBSTITUTION OF THE CODE FOR A CARRIAGE RETURN, OR SIMILAR FUNCTION, WHENEVER A "ZERO BYTE" WAS DETECTED.

IN ADDITION TO THE FORMAT CONSIDERATION WHEN OUTPUTTING TEXT FOR USE WITH OTHER PROGRAMS, ONE MIGHT ALSO WANT TO CONSIDER "TIMING" MATTERS. FOR INSTANCE, SUPPOSE ONE WANTED TO USE A MAGNETIC TAPE STORAGE UNIT FOR SAVING THE TEXT, AND ONE ALSO WAS CONSIDERING USING THE SAME MAGNETIC TAPE SYSTEM TO INPUT A SOURCE LISTING TO AN "ASSEMBLER" PROGRAM. IN MOST TYPICAL APPLICATIONS OF THIS SORT, IT WOULD NOT BE WISE TO SIMPLY DUMP THE INFORMATION IN THE TEXT BUFFER OUT TO THE TAPE UNIT AT THE MAXIMUM RATE AT WHICH THE TAPE SYSTEM CAN OPERATE. THE REASON FOR THIS IS AS FOLLOWS.

MOST TYPICAL "ASSEMBLER" PROGRAMS OPERATE BY "READING" A LINE OF SOURCE CODING AND THEN PERFORMING A VARIETY OF FUNCTIONS DEPENDING ON WHAT WAS CONTAINED IN THE INFORMATION PROCESSED ON THE LINE. THE VARIOUS FUNCTIONS THAT AN ASSEMBLER MIGHT HAVE TO PERFORM MIGHT TAKE A CONSIDERABLE AMOUNT OF TIME (SAY, A FEW TENTHS OF A SECOND) COMPARED TO THE RATE AT WHICH A TAPE UNIT CAN UNLOAD DATA. IF THE TAPE HAD BEEN ORIGINALLY WRITTEN AS ONE LONG CONTINUOUS BLOCK OF TEXT DATA, IT IS QUITE LIKELY THAT AN ASSEMBLER PROGRAM COULD NOT FINISH PROCESSING THE INFORMATION ON ONE LINE BEFORE THE TAPE UNIT WAS ALREADY SENDING INFORMATION ON THE NEXT LINE! A WAY AROUND THE PROBLEM WOULD BE TO SIMPLY PROVIDE A "GAP" ON THE TAPE AFTER EACH LINE OF TEXT. THE LENGTH OF THE GAP SHOULD BE SUFFICIENT TO ALLOW THE MAXIMUM (WORST CASE) CONDITION TO BE PROCESSED BY THE "ASSEMBLER" PROGRAM.

THE SAME KIND OF SITUATION MIGHT APPLY IF A TAPE UNIT WAS USED TO TRANSFER INFORMATION PREPARED ON THE EDITOR PROGRAM TO CONTROL A PIECE OF MACHINERY - SUCH AS AN "NMC" LATHE OR MILLING MACHINE. THE MACHINE MIGHT OPERATE BY READING A LINE OR TWO OF INFORMATION, AND THEN TAKE, QUITE POSSIBLY, MANY SECONDS TO PERFORM A FUNCTION. IN SUCH AN APPLICATION, IT MIGHT BE DESIRABLE TO ACTUALLY PROVIDE A FORMAT ON THE TAPE UNIT SO THAT THE TAPE WAS STARTED AND STOPPED FOR EACH LINE OF TEXT!

A FINAL NOTE ON THE SUBJECT WOULD INCLUDE THE OBSERVATION THAT THE BULK OUTPUT ROUTINE MIGHT BE A GOOD PLACE FOR A USER TO INSERT A CODE CONVERSION ROUTINE. FOR INSTANCE, A USER MIGHT BE EQUIPPED WITH AN ASCII ENCODED ELECTRONIC KEYBOARD AND A "TVT" DISPLAY SYSTEM AND FIND SUCH A SYSTEM IDEAL TO USE WHEN ACTUALLY OPERATING THE EDITOR PROGRAM. HOWEVER, THE USER MIGHT ALSO HAVE A BAUDOT CODE PAPER TAPE PUNCH/READER SYSTEM ON WHICH IT IS DESIRABLE TO SAVE MATERIAL FOR LONG TERM STORAGE. THE USER COULD THEN SIMPLY INSERT AN "ASCII TO BAUDOT" CODE CONVERSION ROUTINE WITHIN THE BULK STORAGE OUTPUT ROUTINE TO TRANSLATE THE TEXT IN THE TEXT BUFFER AS IT IS OUTPUTTED TO THE DEVICE.

NATURALLY, WHATEVER CONSIDERATIONS AND CAPABILITIES ONE PROVIDES FOR THE BULK STORAGE OUTPUT DEVICE, MUST BE PROVIDED IN THE REVERSE MANNER FOR THE BULK STORAGE INPUT ROUTINE WHICH IS CALLED DURING THE OPERATION

OF THE "READ" COMMAND DISCUSSED NEXT.

THE "READ" COMMAND ROUTINE

THE "READ" COMMAND ROUTINE PERFORMS ESSENTIALLY THE REVERSE OF THE "WRITE" COMMAND JUST DESCRIBED. IT CALLS A USER PROVIDED BULK STORAGE INPUT ROUTINE WHICH IT EXPECTS TO LOAD TEXT MATERIAL FROM THE BULK STORAGE DEVICE INTO THE MAIN TEXT BUFFER. THE FIRST FOUR BYTES PLACED IN THE MAIN TEXT BUFFER ARE EXPECTED TO BE THE TEXT BUFFER POINTER AND LINE COUNTER. (REMEMBER TOO, THAT LINES IN THE MAIN TEXT BUFFER ARE EXPECTED TO BE TERMINATED BY A "ZERO BYTE." THUS, IF THE USER PROVIDED CAPABILITIES AS DESCRIBED IN THE PREVIOUS SECTION THAT ALTER THE LINE TERMINATOR, THE "READ" ROUTINE PROVIDED BY THE USER SHOULD PERFORM THE REVERSE PROCEDURE AND RESTORE THE "ZERO BYTE" AS THE LINE TERMINATOR!)

WHEN THE USER PROVIDED BULK INPUT ROUTINE HAS FINISHED, THE ROUTINE SHOWN BELOW PROCEEDS TO TRANSFER THE INFORMATION IN THE FIRST FOUR BYTES OF THE MAIN TEXT BUFFER TO THEIR WORKING LOCATIONS ON PAGE 00 SO THAT THE EDITOR PROGRAM IS "INITIALIZED" FOR THE NEW CONTENTS OF THE TEXT BUFFER WHICH HAVE JUST BEEN LOADED. THE ROUTINE THEN RETURNS CONTROL TO THE EDITOR COMMAND INTERPRETOR ROUTINE.

```
    MNEMONIC                COMMENTS
 --------------          --------------

/
EREAD, CAL READ      /READ IN FROM BULK STORAGE
LHI 007              /NO, SET PNTR TO START OF THE
LLI 000              /TEXT BFR AREA
CAL LDBE             /FETCH TEXT BFR PNTR & LINE NO.
LHI 000              /SET PNTR TO TABLE AREA ON PG 00
LLI 160
CAL SVBE             /STORE TEXT BFR PNTR & LINE NO.
JMP INCMD            /RETURN TO COMMAND MODE
/
```

PUTTING IT ALL TOGETHER - THE ASSEMBLED EDITOR PROGRAM

THAT IS ALL THERE IS TO IT! THE ROUTINES PRESENTED ARE SHOWN NEXT PACKAGED TOGETHER AS AN ASSEMBLED LISTING. THESE ROUTINES, ALONG WITH THE USER PROVIDED I/O ROUTINES DISCUSSED MAY BE LOADED INTO A USER'S SYSTEM AND THE USER MAY PROCEED TO ENJOY THE MANY BENEFITS OF AN EDITOR PROGRAM!

THE FIRST PART OF THE LISTING SHOWS THE LOCATIONS USED ON PAGE 00. MUCH OF THE LOWER PART OF THE PAGE IS USED FOR TEMPORARY STORAGE OF VARIOUS POINTERS AND COUNTERS. AS THE READER HAS BEEN INFORMED, THE UPPER HALF OF PAGE 00 IS RESERVED FOR USE AS THE TEXT INPUT BUFFER.

THE VARIOUS ROUTINES THAT HAVE BEEN DESCRIBED ARE LOCATED STARTING ON PAGE 01 THROUGH PAGE 05 WITH PAGE 06 RESERVED FOR USER PROVIDED I/O ROUTINES. THE EXPECTED STARTING ADDRESSES OF THE USER I/O ROUTINES ARE INDICATED AT THE END OF THE ASSEMBLED LISTING.

THE USER SHOULD NOTE THAT THE STARTING ADDRESS (TO PUT THE PROGRAM IN OPERATION) IS ON PAGE 01 AT LOCATION 033.

```
                        /
000 130   215           215     /MESSAGE TABLE - CARRIAGE RETURN
000 131   212           212     /LINE FEED
000 132   257   276     257     /">"
000 133   000           000     /END OF MSG INDICATOR
000 134   215           215     /CARRIAGE RETURN
000 135   212           212     /LINE FEED
000 136   000           000     /END OF MESSAGE BYTE
                        /
000 137   000           000     /AVAILABLE
                        /
000 140   000           000     /TEMPORARY CPU REGISTER STORAGE
000 141   000           000
000 142   000           000
000 143   000           000
000 144   000           000     /SEARCH INFORMATION STORAGE
000 145   000           000
000 146   000           000
000 147   000           000
000 150   000           000     /DIGIT STORAGE
000 151   000           000     /FOR OCTAL NUMBER
000 152   000           000     /ROUTINE
                        /
000 153   000           000     /AVAILABLE
000 154   000           000     /AVAILABLE
                        /
000 155   104           104     /COMMAND ROUTINE JUMP INSTRUCTION
000 156   000           000     /DIRECTED JUMP
000 157   000           000     /ADDRESS
                        /
000 160   000           000     /TEXT BUFFER PNTR LOW ADDR
000 161   000           000     /TEXT BUFFER PNTR PG ADDR
000 162   000           000     /LINE NUMBER - LOW VALUE
000 163   000           000     /LINE NUMBER - HIGH VALUE
000 164   000           000     /TEMP LINE COUNT - LOW VALUE
000 165   000           000     /TEMP LINE COUNT - HIGH VALUE
000 166   000           000     /LOW LIMIT - LOW VALUE
000 167   000           000     /LOW LIMIT - HIGH VALUE
000 170   000           000     /HIGH LIMIT - LOW VALUE
000 171   000           000     /HIGH LIMIT - HIGH VALUE
000 172   000           000     /LAST PAGE OF TEXT BUFFER
000 173   000           000     /TEMPORARY STORAGE LOCATION
                        /
000 174   000           000     /AVAILABLE
000 175   000           000     /AVAILABLE
                        /
000 176   000           000     /TEMPORARY STORAGE LOCATION
000 177   000           000
                        /
000 200   000           000     /LOC 200 UP RESERVED FOR TX INPUT BFR
                        /
                        /COMMAND LOOK UP TABLE
                        /
001 000   301           301             /APPEND
001 001   025           025
001 002   003           003
001 003   303           303             /CHANGE
001 004   011           011
001 005   005           005
```

```
001 006   304           304                  /DELETE
001 007   242           242
001 010   003           003
001 011   311           311                  /INSERT
001 012   164           164
001 013   004           004
001 014   313           313                  /KILL
001 015   175           175
001 016   003           003
001 017   314           314                  /LIST
001 020   365           365
001 021   003           003
001 022   322           322                  /READ
001 023   122           122
001 024   003           003
001 025   323           323                  /SEARCH
001 026   022           022
001 027   005           005
001 030   327           327                  /WRITE
001 031   146           146
001 032   003           003
                        /
001 033   056 000       INCMD, LHI 000 ——    /SET PNTR TO C/R,L/F,>
001 035   066 130       LLI 130
001 037   106 276 001   CAL MSG              /PRINT COMMAND HEADING
001 042   106 123 001   CAL CDIN             /INPUT THE COMMAND
001 045   307           LAM                  /FETCH THE 1ST CHAR
001 046   036 011       LDI 011              /SET NO. OF CMND CNTR
001 050   365           LLH                  /SET CMND TABLE PNTR
001 051   056 001       LHI 001 ——
001 053   277           LKCMD, CPM           /IS INP = CMND CHAR?
001 054   150 101 001   JTZ FOUND            /YES,PROCESS CMND
001 057   060           INL                  /NO, ADV. CMND TBL PNTR
001 060   060           INL
001 061   060           INL
001 062   031           DCD                  /LAST CMND CHAR CHECKED?
001 063   110 053 001   JFZ LKCMD            /NO, TRY AGAIN
                        /                     YES, PRINT ERR MSG
001 066   106 264 001   ERR, CAL HDLN        /PRINT C/R,L/F
001 071   006 311       LAI 311              /SET LETTER 'I' FOR OUTPUT
001 073   106 200 006   ERP, CAL PRINT       /PRINT ERR MSG
001 076   104 033 001   JMP INCMD            /RETURN TO CMND MODE WITHOUT
                        /
001 101   060           FOUND, INL           /ADV CMND PNTR TO ADDR OF CMND
001 102   307           LAM                  /FETCH LO ADDR OF CMND
001 103   060           INL
001 104   337           LDM                  /FETCH PG ADDR OF CMND
001 105   066 156       LLI 156              /SET PNTR TO JUMP INSTRUCTION
001 107   056 000       LHI 000 ——
001 111   370           LMA                  /SET LO ADDR OF CMND
001 112   060           INL
001 113   373           LMD                  /SET PG ADDR OF CMND
001 114   364           LLE                  /SET PNTR TO END OF CMND INP
001 115   104 155 000   JMP 155 000          /JUMP TO THE CMND ROUTINE
                        /
001 120   106 264 001   BDLN, CAL HDLN       /PRINT C/R,L/F
001 123   106 256 001   CDIN, CAL FBFLM      /FETCH INP BFR LIMIT
001 126   361           LLB                  /SET INP BFR PNTR
001 127   006 240       LAI 240              /FILL INP BFR WITH SPACES
```

```
001 131  370          SP1, LMA
001 132  060          INL                  /IS ENTIRE BFR FULL?
001 133  110 131 001  JFZ SP1              /NO, CONTINUE
001 136  361          LLB                  /YES, SET INP BFR PNTR
001 137  106 000 006  IN2, CAL RCV         /INP A CHAR
001 142  074 223      CPI 223              /INP = CNT'L S?
001 144  150 120 001  JTZ BDLN             /YES, DELETE LINE INPUT
001 147  074 204      CPI 204              /INP = CNT'L D?
001 151  150 033 001  JTZ INCMD            /YES, RET TO CMND MODE
001 154  074 211      CPI 211              /INP = TAB?
001 156  150 231 001  JTZ TAB              /YES, DO TAB FUNCTION
001 161  074 216      CPI 216              /INP = CNT'L CHAR < M?
001 163  140 205 001  JTC RTN2             /YES, RET TO CALLING PGM
001 166  074 377      CPI 377              /INP = RUBOUT?
001 170  150 213 001  JTZ BDCR             /YES, DELETE CHAR
001 173  060          INL                  /IS INP BFR FULL?
001 174  061          DCL
001 175  150 137 001  JTZ IN2              /YES. DON'T STORE CHAR
001 200  370          LMA                  /NO, STORE CHAR
001 201  060          INL                  /INCR INP BFR PNTR
001 202  104 137 001  JMP IN2              /GET NEXT CHAR
                       /
001 205  106 256 001  RTN2, CAL FBFLM      /FETCH INP BFR LIMIT
001 210  361          LLB                  /RESET INP BFR PNTR
001 211  307          LAM                  /FETCH 1ST CHAR
001 212  007          RET                  /RETURN TO CALLING PGM
                       /
001 213  106 256 001  BDCR, CAL FBFLM      /FETCH INP BFR LIMIT
001 216  301          LAB
001 217  276          CPL                  /ANY CHAR'S STORED YET?
001 220  150 137 001  JTZ IN2              /NO, GET NEXT CHAR
001 223  061          DCL                  /YES, BACK UP PNTR
001 224  076 240      LMI 240              /STORE A SPACE
001 226  104 137 001  JMP IN2              /GET NEXT CHAR
                       /
001 231  060          TAB, INL             /CHECK FOR INP BFR FULL
001 232  061          DCL
001 233  150 137 001  JTZ IN2              /IF FULL, IGNORE TAB
001 236  006 240      LAI 240              /SET UP SPACE CHAR IN ACC
001 240  370          LMA                  /STORE SPACE IN INP BFR
001 241  106 200 006  CAL PRINT            /PRINT SPACE
001 244  060          INL                  /INCR INP BFR PNTR
001 245  306          LAL
001 246  044 007      NDI 007              /IS TAB OVER TO LIMIT
001 250  150 137 001  JTZ IN2              /YES, INP NEXT CHAR
001 253  104 231 001  JMP TAB              /NO, CONT. WITH SPACES
                       /
001 256  346          FBFLM, LEL           /SAVE INP PNTR
001 257  016 270      LBI 270              /LOAD INP BFR LIMIT
001 261  056 000      LHI 000              /RESTORE INP PNTR
001 263  007          RET                  /RET TO CALLING PGM
                       /
001 264  066 134      HDLN, LLI 134        /SET PNTR TO C/R,L/F MSG
001 266  056 000      LHI 000
001 270  106 276 001  CAL MSG              /PRINT THE C/R,L/F
001 273  056 000      LHI 000              /RESET PAGE PNTR
001 275  007          RET                  /RETURN
                       /
001 276  307          MSG, LAM             /FETCH CHAR TO PRINT
001 277  240          NDA                  /=000?
001 300  053          RTZ                  /YES,DONE. RETURN
```

```
001 301   106 200 006   CAL PRINT          /PRINT THE CHAR.
001 304   106 143 002   CAL INMEM          /INCR MEM PNTR
001 307   104 276 001   JMP MSG            /CONTINUE OUTPUT
                        /
001 312   346           OCTNM, LEL         /SAVE REG L
001 313   106 345 001   CAL OCTPR          /FETCH OCTAL NO. PAIR
001 316   066 166       LLI 166            /SET STORAGE PNTR
001 320   371           LMB                /STORE OCTAL PAIR IN LOC.166
001 321   060           INL                /AND 167 ON PG 00
001 322   372           LMC
001 323   364           LLE
001 324   307           LAM                /FETCH NEXT CHAR
001 325   074 254       CPI 254            /COMMA?
001 327   110 337 001   JFZ SGL            /NO, SINGLE INPUT
001 332   060           INL                /YES, FETCH 2ND OCTAL PAIR
001 333   346           LEL
001 334   106 345 001   CAL OCTPR
001 337   066 170       SGL, LLI 170       /STORE OCTAL PAIR IN LOC. 170
001 341   371           LMB                /AND 171 ON PG 00
001 342   060           INL
001 343   372           LMC
001 344   007           RET                /RETURN
                        /
001 345   106 355 001   OCTPR, CAL DCDNM   /CONVERT INP TO OCT NO.
001 350   321           LCB                /SAVE IN REG C
001 351   040           INE
001 352   104 355 001   JMP DCDNM          /CONVERT 2ND OCT NO. AND RET
                        /
001 355   106 367 001   DCDNM, CAL DCON    /FETCH OCTAL DIGITS
001 360   106 111 002   CAL OCT            /FORM OCTAL NO. IN REG B
001 363   120 066 001   JFS ERR            /IF INVALID, PRINT ERR MSG
001 366   007           RET                /RETURN
                        /
001 367   066 150       DCON, LLI 150      /CLEAR DIGIT STORAGE TABLE
001 371   375           LMH                /BY FILLING WITH 000
001 372   060           INL
001 373   375           LMH
001 374   060           INL
001 375   375           LMH
001 376   364           LLE
001 377   106 036 002   LOOP, CAL FNUM     /CHECK FOR VALID NO.
002 002   160 027 002   JTS CKLNH          /IF NOT, CHECK CHAR COUNT= 0
002 005   307           LAM
002 006   336           LDL
002 007   044 007       NDI 007            /IF VALID, MASK OFF 260
002 011   066 150       LLI 150            /STORE OCTAL NUMBER IN
002 013   317           LBM                /TABLE AT LOC 150 PG 00
002 014   370           LMA                /AND SHIFT OTHER NUMBERS
002 015   060           INL                /UP THRU THE TABLE
002 016   307           LAM
002 017   371           LMB
002 020   060           INL
002 021   370           LMA
002 022   363           LLD                /RESTORE AND INCR INP PNTR
002 023   060           INL
002 024   104 377 001   JMP LOOP           /FETCH NEXT NUMBER
                        /
```

```
002 027  306          CKLNH, LAL
002 030  274          CPE                 /IS CHAR COUNT= 0?
002 031  150 066 001  JTZ ERR             /YES, PRINT ERR MSG
002 034  346          LEL                 /NO, SET REG E AND RETURN
002 035  007          RET
                      /
002 036  307          FNUM, LAM           /IS CHAR A VALID NUMBER?
002 037  074 260      CPI 260
002 041  063          RTS                 /IF NOT, RET WITH S FLAG SET
002 042  024 270      SUI 270
002 044  004 200      ADI 200             /IF SO, RET WITH S FLAG RESET
002 046  007          RET
                      /
002 047  004 001      INCR, ADI 001       /INCR CONTENTS OF MEM LOC
002 051  370          LMA                 /RESTORE MEM LOC
002 052  003          RFC                 /IF NO CARRY, RETURN
002 053  060          INL                 /ELSE, FETCH NEXT LOC
002 054  307          LAM
002 055  004 001      ADI 001             /INCR MEM LOC
002 057  370          LMA                 /RESTORE MEM LOC
002 060  007          RET                 /AND RETURN
                      /
002 061  066 140      LD140, LLI 140      /SET PNTR TO LOC 140 PG 00
002 063  056 000      LHI 000
002 065  317          LDBE, LBM           /LOAD REG B THRU REG E
002 066  060          INL                 /FROM MEMORY
002 067  327          LCM
002 070  060          INL
002 071  337          LDM
002 072  060          INL
002 073  347          LEM
002 074  007          RET                 /RETURN
                      /
002 075  066 140      SV140, LLI 140      /SET PNTR TO LOC 140 PG 00
002 077  056 000      LHI 000
002 101  371          SVBE, LMB           /SAVE REG B THRU REG E
002 102  060          INL                 /IN MEMORY
002 103  372          LMC
002 104  060          INL
002 105  373          LMD
002 106  060          INL
002 107  374          LME
002 110  007          RET                 /RETURN
                      /
002 111  066 152      OCT, LLI 152        /SET PNTR TO 3RD DIGIT
002 113  307          LAM
002 114  074 004      CPI 004             /IS DIGIT > 3?
002 116  023          RFS                 /YES, RET WITH S FLAG RESET
002 117  044 003      NDI 003             /CLEAR CARRY
002 121  012          RRC
002 122  012          RRC                 /POSITION DIGIT
002 123  310          LBA                 /SAVE IN REG B
002 124  061          DCL                 /DECR PNTR
002 125  307          LAM                 /FETCH NEXT DIGIT
002 126  002          RLC
002 127  002          RLC
002 130  002          RLC                 /POSITION DIGIT
002 131  201          ADB                 /ADD TO REG B
002 132  310          LBA                 /SAVE IN REG B
002 133  061          DCL                 /DECR PNTR
```

```
002 134  307          LAM              /FETCH LAST DIGIT
002 135  201          ADB              /ADD TO REG B
002 136  310          LBA              /STORE FINAL NO. IN REG B
002 137  006 200      LAI 200          /SET S FLAG TO INDICATE
002 141  240          NDA              /THAT THE NO. WAS VALID
002 142  007          RET              /AND RETURN
                      /
002 143  060          INMEM, INL       /INCR LO ADDR
002 144  013          RFZ              /RET IF NOT ZERO
002 145  050          INH              /INCR PAGE REG
002 146  007          RET              /RET TO CALLING PGM
                      /
002 147  304          CKOV, LAE        /FETCH INP PNTR
002 150  271          CPB              /IS INP A BLANK LINE?
002 151  110 156 002  JFZ NCHR         /NO, CONTINUE
002 154  040          INE              /YES, SET OUP ONE SPACE
002 155  304          LAE
002 156  011          NCHR, DCB        /DECR INP BFR LIMIT
002 157  221          SUB              /CALC. CHAR COUNT+1
002 160  066 173      LLI 173          /SET PNTR TO TBL AREA
002 162  370          LMA              /SAVE CHAR COUNT+1
002 163  320          LCA              /SAVE CHAR COUNT+1 IN REG C
002 164  066 160      LLI 160          /SET PNTR TO TX BFR PNTR
002 166  010          INB              /INCR INP BFR LIMIT
002 167  207          ADM              /ADD CC+1 TO TX BFR PNTR
002 170  003          RFC              /RET ON NO CARRY
002 171  060          INL              /FETCH TX BFR PAGE PNTR
002 172  307          LAM
002 173  066 172      LLI 172          /SET PNTR TO PAGE LIMIT
002 175  277          CPM              /IS CUR PAGE = LAST AVAIL?
002 176  150 272 002  JTZ OFL          /YES,PRINT OVERFLOW ERROR
002 201  066 160      LLI 160          /NO, SET PNTR TO TX BFR PNTR
002 203  007          RET              /RETURN TO CALLING PGM
                      /
                      /
002 204  353          ZLOK, LHD        /SET PNTR TO TX BFR
002 205  364          LLE
002 206  307          ZL1, LAM         /FETCH CHAR FROM TX BFR
002 207  106 143 002  CAL INMEM        /INCR TEXT BFR PNTR
002 212  240          NDA              /IS CHAR = 000?
002 213  110 206 002  JFZ ZL1          /NO, TRY NEXT CHAR
002 216  346          LEL              /YES, SAVE TX BFR PNTR
002 217  335          LDH
002 220  007          RET              /RETURN TO CALLING PGM
                      /
002 221  046 004      FND, LEI 004     /SET START OF TX BFR PNTR
002 223  036 007      LDI 007
002 225  066 164      LLI 164          /SET PNTR TO LINE COUNT
002 227  056 000      LHI 000
002 231  076 001      LMI 001          /SET LINE COUNT TO 000 001
002 233  060          INL
002 234  375          LMH
002 235  066 167      FD1, LLI 167     /SET PNTR TO LO LIMIT
002 237  307          LAM              /FETCH LO LIMIT
002 240  066 165      LLI 165          /SET PNTR TO LINE COUNT
002 242  277          CPM              /IS LINE COUNT = LO LIMIT?
002 243  110 254 002  JFZ FD2          /NO,FIND START OF NEXT LINE
002 246  060          INL              /CHECK OTHER HALF OF COUNT
002 247  307          LAM
```

```
002 250   061          DCL
002 251   061          DCL
002 252   277          CPM                 /IS LINE COUNT = LO LIMIT
002 253   053          RTZ                 /YES, RET TO CALLING PGM
002 254   106 204 002  FD2, CAL ZLOK       /NO, SEARCH FOR NEXT LINE
002 257   056 000      LHI 000             /SET PNTR TO LINE COUNT
002 261   066 164      LLI 164
002 263   307          LAM                 /FETCH LINE COUNT
002 264   106 047 002  CAL INCR            /INCR LINE COUNT
002 267   104 235 002  JMP FD1             /CONTINUE
                       /
002 272   106 264 001  OFL, CAL HDLN       /PRINT C/R, L/F
002 275   006 302      LAI 302             /SET UP OVERFLOW ERROR CODE
002 277   104 073 001  JMP ERP             /PRINT OVERFLOW ERROR
                       /
                       /
002 302   353          SHFT, LHD           /SET PNTR TO 'FROM' LOC
002 303   364          LLE
002 304   307          LAM                 /FETCH CHAR TO TRANSFER
002 305   351          LHB                 /SET PNTR TO 'TO' LOC
002 306   362          LLC
002 307   370          LMA                 /STORE CHAR IN NEW LOC
002 310   007          RET                 /RET TO CALLING PGM
                       /
002 311   020          NCR, INC            /INCR 'TO' LO ADDR
002 312   110 316 002  JFZ NCRD            /IF NOT 0, SKIP NEXT
002 315   010          INB                 /INCR 'TO' PAGE
002 316   040          NCRD, INE           /INCR 'FROM' LO ADDR
002 317   013          RFZ                 /IF NOT 0, RET
002 320   030          IND                 /INCR 'FROM' PAGE
002 321   007          RET                 /RET TO CALLING PGM
                       /
002 322   066 166      FLO, LLI 166        /SET PNTR TO LO LIMIT
002 324   056 000      LHI 000
002 326   307          LAM                 /FETCH LO LIMIT
002 327   240          NDA                 /IS 1ST HALF = 000?
002 330   013          RFZ                 /NO, RET WITH Z FLAG RESET
002 331   060          INL                 /YES, FETCH 2ND HALF
002 332   307          LAM
002 333   240          NDA                 /SET UP Z FLAG
002 334   007          RET                 /RET TO CALLING PGM
                       /
                       /
002 335   066 162      FLINO, LLI 162      /SET PNTR TO LINE NO.
002 337   056 000      LHI 000
002 341   307          LAM                 /FETCH LINE NO.
002 342   240          NDA                 /SET UP FLAGS
002 343   007          RET                 /RET TO CALLING PGM
                       /
002 344   024 001      DECR, SUI 001       /DECR VALUE IN ACC
002 346   370          LMA                 /STORE VALUE BACK IN MEM
002 347   003          RFC                 /RET IF NO CARRY
002 350   060          INL                 /ELSE, SET PNTR TO NX LOC
002 351   307          LAM                 /FETCH 2ND HALF OF VALUE
002 352   024 001      SUI 001             /DECR 2ND HALF
002 354   370          LMA                 /STORE VALUE BACK IN MEM
002 355   007          RET                 /RET TO CALLING PGM
                       /
```

```
002 356  361          LDHILO, LLB        /SET PNTR TO S.A. OF INP BFR
002 357  060          INL                /MOVE PNTR TO 3RD CHAR
002 360  060          INL
002 361  106 312 001  CAL OCTNM          /FETCH NMBR'S FROM INP BFR
002 364  066 166      LLI 166            /SET PNTR TO LO LIMIT
002 366  106 065 002  CAL LDBE           /FETCH LO & HI LIMIT
002 371  304          LAE                /COMPARE LO LIMIT TO HI LIMIT
002 372  272          CPC
002 373  140 066 001  JTC ERR            /IF LO > HI, THEN ERROR
002 376  110 006 003  JFZ CKLI           /IF NOT =, THEN SKIP NEXT
003 001  303          LAD                /CHECK 2ND HALF LO > HI
003 002  271          CPB
003 003  140 066 001  JTC ERR            /IF LO > HI, PRINT ERROR MSG
003 006  066 163      CKLI, LLI 163      /SET PNTR TO LINE NO.
003 010  307          LAM                /COMPARE HI LIMIT TO LINE NO.
003 011  274          CPE
003 012  140 066 001  JTC ERR            /IF HI > LN NO., ERROR
003 015  013          RFZ                /RET IF NOT EQUAL
003 016  061          DCL                /CHECK 2ND HALF HI > LN NO.
003 017  307          LAM
003 020  273          CPD
003 021  140 066 001  JTC ERR            /IF HI > LN NO., ERROR
003 024  007          RET                /ELSE, RET TO CALLING PGM
                       /
003 025  106 264 001  APND, CAL HDLN      /PRINT C/R,L/F
003 030  106 264 001  A1, CAL HDLN        /PRINT C/R, L/F
003 033  106 123 001  CAL CDIN           /INPUT LINE OF TEXT
003 036  106 147 002  CAL CKOV           /CHECK POSSIBLE BUFF OVRFLO
003 041  334          LDE                /SAVE INP BFR PNTR
003 042  301          LAB
003 043  327          LCM                /FETCH TEXT BFR PNTR
003 044  060          INL
003 045  317          LBM
003 046  360          LLA                /SET UP S.A. OF INP BFR
003 047  056 000      A2, LHI 000
003 051  307          LAM                /FETCH CHAR FROM INP BFR
003 052  346          LEL
003 053  351          LHB                /SET UP TEXT BFR PNTR
003 054  362          LLC
003 055  370          LMA                /STORE CHAR IN TEXT BFR
003 056  106 143 002  CAL INMEM           /INCR TEXT BFR PNTR
003 061  315          LBH                /SAVE TEXT BFR PNTR
003 062  326          LCL
003 063  040          INE                /INCR INP BFR PNTR
003 064  304          LAE                /MOVE INP BFR PNTR TO ACC
003 065  273          CPD                /LAST INP CHAR MOVED TO TX BFR
003 066  150 075 003  JTZ A3             /YES, WRAP UP LINE APPEND
003 071  364          LLE                /NO, SET INP BFR PNTR
003 072  104 047 003  JMP A2             /STORE MORE TEXT
003 075  250          A3, XRA            /STORE THE END OF TEXT CHAR
003 076  370          LMA                /IN THE TEXT BFR
003 077  020          INC                /INCR TX BFR PNTR LO ADDR
003 100  110 104 003  JFZ A4             /IF NOT 0, SKIP TO A4
003 103  010          INB                /ELSE INCR PAGE PNTR
003 104  350          A4, LHA            /SET PNTR TO TX BF PNTR STRAGE
003 105  066 160      LLI 160
003 107  372          LMC                /SAVE NEW TX BFR PNTR LO ADDR
003 110  060          INL
003 111  371          LMB                /AND PAGE ADDR
003 112  060          INL                /SET PNTR TO LINE NUMBER
```

```
003 113   307          LAM                  /FETCH 1ST HALF OF LINE NUMBER
003 114   106 047 002  CAL INCR             /INCR LINE NUMBER
003 117   104 030 003  JMP A1               /SET UP FOR NEW LINE TO APPEND
                       /
                       /READ ROUTINE
                       /
003 122   106 152 006  EREAD, CAL READ      /READ IN FROM BULK STORAGE
003 125   056 007      LHI 007              /NO, SET PNTR TO START OF THE
003 127   066 000      LLI 000              /TEXT BFR AREA
003 131   106 065 002  CAL LDBE             /FETCH TEXT BFR PNTR & LINE
003 134   056 000      LHI 000              /SET PNTR TO TABLE AREA ON PG0
003 136   066 160      LLI 160
003 140   106 101 002  CAL SVBE             /STORE TEXT BFR PNTR & LINE
003 143   104 033 001  JMP INCMD            /RETURN TO COMMAND MODE
                       /
                       /WRITE ROUTINE
                       /
003 146   066 160      WRITE, LLI 160       /SET PNTR TO TX BF PNTR & LIN
003 150   056 000      LHI 000
003 152   106 065 002  CAL LDBE             /FETCH TX BF PNTR & LINE NO.
003 155   365          LLH                  /SET PNTR TO START OF TEXT BF
003 156   056 007      LHI 007
003 160   106 101 002  CAL SVBE             /STORE TEXT BF PNTR & LINE NO.
003 163   066 000      LLI 000              /TO BE WRITTEN TO BULK STORAGE
003 165   332          LDC                  /SET START AND END ADDR FOR
003 166   341          LEB                  /LIMITS OF TX BFR TO BE STORED
003 167   106 300 006  CAL PUNCH            /WRITE TX BFR TO BULK STORAGE
003 172   104 033 001  JMP INCMD            /RETURN TO COMMAND MODE
                       /
                       /KILL ROUTINE
                       /
003 175   056 000      KILL, LHI 000        /SET PNTR TO TABLE AREA
003 177   066 160      LLI 160
003 201   076 004      LMI 004              /SET TX BFR PNTR TO START
003 203   060          INL                  /OF THE TX BFR AREA
003 204   076 007      LMI 007
003 206   060          INL                  /SET PNTR TO LINE NO.
003 207   375          LMH                  /SET LINE NO. TO 000 000
003 210   060          INL
003 211   375          LMH
003 212   365          LLH                  /SET PNTR TO THE LAST POSSIBLE
003 213   056 077      LHI 077              /PAGE OF MEMORY
003 215   317          CKE, LBM             /SAVE MEM CONTENTS
003 216   376          LML                  /WRITE 000 TO MEM LOC
003 217   307          LAM                  /READ IT BACK
003 220   371          LMB                  /RESTORE MEM CONTENTS
003 221   240          NDA                  /IS MEMORY AVAILABLE?
003 222   150 231 003  JTZ ENM              /YES, SET MAX PAGE LIMIT
003 225   051          DCH                  /NO, DECR PAGE PNTR
003 226   104 215 003  JMP CKE              /CHECK NEXT PAGE
003 231   315          ENM, LBH             /SAVE PAGE NUMBER
003 232   066 172      LLI 172              /SET PNTR TO TABLE AREA TO
003 234   056 000      LHI 000              /STORE LIMIT OF TX BFR
003 236   371          LMB
003 237   104 033 001  JMP INCMD            /RETURN TO COMMAND MODE
                       /
```

```
003 242   106 356 002   DELET, CAL LDHILO     /SET LINE NUMBERS FROM CMND
003 245   106 253 003   CAL DLET             /DELETE LINES SPECIFIED
003 250   104 033 001   JMP INCMD            /RETURN TO COMMAND MODE
                        /
003 253   106 322 002   DLET, CAL FLO        /IS LO LIMIT = 0?
003 256   150 066 001   JTZ ERR              /YES, PRINT ERROR MSG
003 261   106 221 002   CAL FND              /FIND FIRST LINE TO DELETE
003 264   324           LCE                  /SAVE START ADDRESS
003 265   313           LBD
003 266   106 204 002   D3, CAL ZLOK         /FIND END OF LINE
003 271   106 335 002   CAL FLINO            /FETCH LINE NUMBER
003 274   106 344 002   CAL DECR             /DECR LINE NUMBER
003 277   066 165       LLI 165              /SET PNTR TO LINE COUNT
003 301   307           LAM                  /FETCH LINE COUNT
003 302   066 171       LLI 171              /SET PNTR TO HI LINE LIMIT
003 304   277           CPM                  /IS FINAL LINE TO DLT RCHED
003 305   110 320 003   JFZ D4               /NO, CONTINUE DELETE
003 310   061           DCL                  /IF 1ST HALF OF LN NO. MATCH
003 311   307           LAM                  /CHECK FOR MATCH OF OTHER HALF
003 312   066 164       LLI 164
003 314   277           CPM                  /IS FINAL LINE REACHED?
003 315   150 331 003   JTZ SDLT             /YES, COMPLETE DELETE MODE
003 320   066 164       D4, LLI 164          /NO, SET PNTR TO LINE COUNT
003 322   307           LAM
003 323   106 047 002   CAL INCR             /INCR LINE COUNT
003 326   104 266 003   JMP D3               /DELETE MORE LINES
003 331   066 160       SDLT, LLI 160        /SET PNTR TO START OF TBL AREA
003 333   307           LAM                  /FETCH CURRENT TX BFR PNTR
003 334   274           CPE                  /IS END OF TEXT BFR REACHED?
003 335   110 346 003   JFZ D6               /NO, SHIFT MORE TX DOWN IN BFR
003 340   060           INL                  /CHECK 2ND HALF OF PNTR'S
003 341   307           LAM                  /FOR A MATCH
003 342   273           CPD                  /DOES 2ND HALF MATCH?
003 343   150 361 003   JTZ D5               /YES,WRAP UP DELETE MODE
003 346   106 302 002   D6, CAL SHFT         /SHIFT CHAR DOWN IN TX BFR
003 351   106 311 002   CAL NCR              /INCR PNTR'S
003 354   056 000       LHI 000              /RESET TABLE PNTR
003 356   104 331 003   JMP SDLT             /CHECK FOR END OF SHIFT
003 361   371           D5, LMB              /SAVE NEW TX BFR PNTR
003 362   061           DCL                  /IN THE TABLE AREA
003 363   372           LMC
003 364   007           RET                  /RETURN TO CALLING PGM
                        /
003 365   061           LIST, DCL            /DECR INP PNTR
003 366   306           LAL
003 367   271           CPB                  /LIST ALL OF TX BFR?
003 370   150 007 004   JTZ LALL             /YES, SET MAX LIMITS
003 373   106 356 002   CAL LDHILO           /NO, SET LINE NMBRS FROM CMND
003 376   106 322 002   CAL FLO              /IS LO LIMIT = 0?
004 001   150 066 001   JTZ ERR              /YES, PRINT ERROR MSG
004 004   104 021 004   JMP LST              /START LIST ROUTINE
                        /
004 007   066 166       LALL, LLI 166        /SET PNTR TO LOW LIMIT STRAGE
004 011   076 001       LMI 001              /SET START OF LIST TO LINE 1
004 013   060           INL
004 014   375           LMH
004 015   060           INL
004 016   375           LMH                  /SET HIGH LIMIT TO 000 000
004 017   060           INL                  /WHICH WILL BE CHANGED TO MAX
004 020   375           LMH                  /LINE NO. LATER
```

```
004 021   106 335 002   LST, CAL FLINO      /IS LINE NO. = 0?
004 024   110 035 004   JFZ LT1             /NO, CONTINUE
004 027   060           INL                 /CHECK 2ND HALF OF LINE NO.
004 030   307           LAM
004 031   240           NDA                 /IS 2ND HALF = 0?
004 032   150 066 001   JTZ ERR             /YES, PRINT ERROR MSG
004 035   066 170       LT1, LLI 170        /SET PNTR TO HI LIMIT
004 037   307           LAM                 /FETCH HI LINE LIMIT
004 040   240           NDA                 /HI LIMIT = 0?
004 041   110 066 004   JFZ LT2             /NO, START LIST OUTPUT
004 044   060           INL                 /CHECK 2ND HALF OF HI LIMIT
004 045   307           LAM
004 046   240           NDA                 /2ND HALF = 0?
004 047   110 066 004   JFZ LT2             /NO, START LIST OUTPUT
004 052   066 162       LLI 162             /YES, SET PNTR TO LINE NO.
004 054   307           LAM                 /FETCH 1ST HALF OF LINE NO.
004 055   066 170       LLI 170             /SET PNTR TO HI LIMIT
004 057   370           LMA                 /SET HI LIMIT TO LINE NO.
004 060   066 163       LLI 163
004 062   307           LAM                 /FETCH 2ND HALF OF LINE NO.
004 063   066 171       LLI 171             /PNTR TO 2ND HALF OF HI LIMIT
004 065   370           LMA                 /SET HI LIMIT TO LINE NO.
004 066   106 264 001   LT2, CAL HDLN       /OUTPUT 2 C/R,L/F
004 071   106 264 001   CAL HDLN
004 074   106 221 002   CAL FND             /FIND START ADDR OF LINE
004 077   353           LT3, LHD            /SET PNTR TO START OF LINE
004 100   364           LLE
004 101   106 276 001   CAL MSG             /PRINT LINE OF TEXT
004 104   106 143 002   CAL INMEM           /INCR TEXT PNTR
004 107   335           LDH                 /SAVE TEXT PNTR
004 110   346           LEL
004 111   056 000       LHI 000             /SET PNTR TO LO LIMIT
004 113   066 167       LLI 167
004 115   307           LAM                 /FETCH LO LIMIT
004 116   066 171       LLI 171             /SET PNTR TO HI LIMIT
004 120   277           CPM                 /IS LO LIMIT = HI LIMIT?
004 121   110 134 004   JFZ LT4             /NO, CONTINUE
004 124   061           DCL                 /CHECK 2ND HALF OF LO & HI LIM
004 125   307           LAM
004 126   066 166       LLI 166
004 130   277           CPM                 /IS LAST LINE LISTED?
004 131   150 033 001   JTZ INCMD           /YES, RET TO CMND MODE
004 134   066 166       LT4, LLI 166        /SET PNTR TO LO LIMIT
004 136   307           LAM                 /FETCH LO LIMIT
004 137   106 047 002   CAL INCR            /INCR LO LIMIT
004 142   066 176       LLI 176             /SET PNTR TO TMP STORAGE
004 144   374           LME                 /STORE TX PNTR IN TMP STORAGE
004 145   060           INL
004 146   373           LMD
004 147   106 264 001   CAL HDLN            /PRINT C/R, L/F
004 152   066 176       LLI 176             /SET PNTR TO TMP STORAGE
004 154   056 000       LHI 000
004 156   347           LEM                 /FETCH TEXT BFR PNTR
004 157   060           INL
004 160   337           LDM
004 161   104 077 004   JMP LT3             /CONTINUE PRINTING
                        /
```

```
004 164   106 356 002   INSRT, CAL LDHILO    /SET LINE NUMBERS FROM CMND
004 167   106 175 004   INS1, CAL NSRT       /INSERT LINE INTO TEXT BFR
004 172   104 167 004   JMP INS1             /SET UP FOR ANOTHER INSERT
                        /
004 175   106 322 002   NSRT, CAL FLO
004 200   150 066 001   JTZ ERR              /YES, PRINT ERROR MSG
004 203   106 264 001   CAL HDLN             /PRINT C/R, L/F
004 206   106 123 001   CAL CDIN             /INPUT NEW LINE TO INSERT
004 211   106 147 002   CAL CKOV             /CHECK FOR POSSIBLE BFR OVRFLO
004 214   106 221 002   CAL FND              /FIND START OF LINE
004 217   066 176       LLI 176              /SET PNTR TO TMP STORAGE
004 221   374           LME                  /SAVE START ADDR OF LINE
004 222   060           INL
004 223   373           LMD
004 224   066 160       SNST, LLI 160        /SET PNTR TO TX BF PNTR
004 226   307           LAM                  /FETCH TX BFR PNTR
004 227   060           INL                  /SET REG D & E AS 'FROM PNTR'
004 230   337           LDM
004 231   340           LEA                  /SET UP END OF LINE PNTR
004 232   202           ADC                  /BY ADDING CHAR COUNT + 1
004 233   320           LCA                  /SET REG B & C AS 'TO PNTR'
004 234   313           LBD
004 235   100 242 004   JFC NS1              /INCR 'TO' PAGE PNTR? NO
004 240   010           INB                  /YES
004 241   371           LMB                  /SET NEW TX BFR PNTR
004 242   061           NS1, DCL
004 243   372           LMC
004 244   106 376 004   CAL CKTX             /'FROM' = END OF TX BFR?
004 247   150 311 004   JTZ NST1             /YES, INSERT LINE
004 252   304           NS2, LAE             /NO, DECR 'FROM'
004 253   024 001       SUI 001
004 255   340           LEA                  /DOES 2ND HALF NEED DECR
004 256   100 262 004   JFC NS3              /NO, SKIP DECR
004 261   031           DCD                  /YES, DECR 2ND HALF OF 'FROM'
004 262   302           NS3, LAC             /DECR 'TO'
004 263   024 001       SUI 001
004 265   320           LCA                  /DOES 2ND HALF NEED DECR
004 266   100 272 004   JFC NS4              /NO, SKIP DECR
004 271   011           DCB                  /YES, DECR 2ND HALF
004 272   106 376 004   NS4, CAL CKTX        /'FROM' = START OF INSERT LOC?
004 275   150 306 004   JTZ NSTL             /YES, INSERT LINE
004 300   106 302 002   CAL SHFT             /NO, SHFT CHAR IN TX BFR
004 303   104 252 004   JMP NS2              /CONTINUE SHIFTING
                        /
004 306   106 302 002   NSTL, CAL SHFT       /SHFT CHAR BEFORE NEW LINE
004 311   066 173       NST1, LLI 173        /SET PNTR TO CHAR COUNT+1
004 313   056 000       LHI 000              /FETCH CHAR COUNT+1
004 315   307           LAM
004 316   024 001       SUI 001              /DECR TO GET CHAR COUNT
004 320   066 176       LLI 176              /SET PNTR TO TMP STORAGE
004 322   207           ADM                  /ADD S.A. TO CHAR COUNT
004 323   370           LMA                  /SAVE FINAL ADDR
004 324   324           LCE                  /SET 'TO PNTR'
004 325   303           LAD
004 326   106 256 001   CAL FBFLM            /FETCH START OF INP BFR
004 331   335           LDH                  /SET 'FROM PNTR'
004 332   341           LEB
004 333   310           LBA                  /SAVE FINAL PAGE IN REG B
004 334   106 302 002   NS6, CAL SHFT        /SHIFT INP TO TX BFR
004 337   106 311 002   CAL NCR              /INCR 'TO' AND 'FROM'
```

```
004 342  066 176          LLI 176              /SET PNTR TO TMP STORAGE
004 344  056 000          LHI 000
004 346  307               LAM                  /FETCH FINAL LOC IN TX BFR
004 347  272               CPC                  /END OF LINE INPUT?
004 350  110 334 004       JFZ NS6              /NO, CONT. STORAGE
004 353  362               LLC                  /YES, SET END OF LINE PNTR
004 354  351               LHB
004 355  076 000          LMI 000              /STORE EOL INDICATOR
004 357  066 166          LLI 166              /SET PNTR TO LO LIMIT
004 361  056 000          LHI 000
004 363  307               LAM                  /FETCH LO LIMIT
004 364  106 047 002       CAL INCR             /INCR LO LIMIT
004 367  066 162          LLI 162              /SET PNTR TO LINE NO.
004 371  307               LAM                  /FETCH LINE NO.
004 372  106 047 002       CAL INCR             /INCR LINE NO.
004 375  007               RET                  /RETURN TO CALLING PGM
                           /
004 376  066 176          CKTX, LLI 176        /SET PNTR TO TMP STORAGE
005 000  056 000          LHI 000
005 002  304               LAE
005 003  277               CPM                  /TMP STORAGE = 'FROM'?
005 004  013               RFZ                  /NO,RET WITH Z FLAG RESET
005 005  060               INL
005 006  303               LAD                  /CHECK 2ND HALF OF ADDR
005 007  277               CPM
005 010  007               RET                  /IF =,RET WITH Z FLAG SET
                           /
005 011  106 356 002       CHANGE, CAL LDHILO   /SET LINE NUMBERS FROM CMND
005 014  106 253 003       CAL DLET             /DELETE LINES SPECIFIED
005 017  104 167 004       JMP INS1             /INSERT NEW LINES
                           /
                           /SEARCH ROUTINE
                           /
005 022  106 356 002       SRCH, CAL LDHILO     /SET LINE NUMBERS FROM CMND
005 025  106 322 002       CAL FLO              /IS LO LIMIT = 0?
005 030  150 066 001       JTZ ERR              /YES, PRINT ERROR MSG
005 033  106 221 002       CAL FND              /FIND S.A. OF LINE
005 036  324               LCE                  /SAVE START ADDR
005 037  313               LBD
005 040  066 176          LLI 176              /SET PNTR TO TMP STORAGE
005 042  056 000          LHI 000
005 044  374               LME                  /SAVE S.A. IN TMP STORAGE
005 045  060               INL
005 046  373               LMD
005 047  364               LLE                  /SAVE S.A. IN REG'S H & L
005 050  353               LHD
005 051  106 000 006       CAL RCV              /INP CHAR TO SEARCH FOR
005 054  335               LDH                  /RESTORE S.A.
005 055  106 256 001       CAL FBFLM            /FETCH S.A. OF INP BFR
005 060  321               LCB                  /SAVE S.A. OF INP BFR
005 061  310               LBA                  /SAVE SEARCH CHAR
005 062  066 144          LLI 144              /SAVE REG'S IN SEARCH TBL
005 064  106 101 002       CAL SVBE
005 067  106 264 001       CAL HDLN             /PRINT C/R,L/F
005 072  066 144          LLI 144
005 074  106 065 002       CAL LDBE             /RESTORE REG'S
005 077  364               S1, LLE              /SET PNTR TO TEXT LINE
005 100  353               LHD
005 101  307               LAM                  /FETCH CHAR FROM TEXT
005 102  240               NDA                  /SHAR = EOL INDICATOR?
```

```
005 103   150 311 005   JTZ SCT          /YES, STORE LINE IN TX BFR
005 106   362           LLC              /SET PNTR TO INP BFR
005 107   056 000       LHI 000
005 111   370           LMA              /STORE CHAR IN INP BFR
005 112   066 144       LLI 144
005 114   106 101 002   CAL SVBE         /SAVE REG'S
005 117   106 200 006   CAL PRINT        /PRINT CHAR
005 122   066 144       LLI 144
005 124   106 065 002   CAL LDBE         /RESTORE REG'S
005 127   106 316 002   CAL NCRD         /INCR TEXT PNTR
005 132   020           INC              /INCR INP BFR PNTR
005 133   150 311 005   JTZ SCT          /=0? YES, STORE LINE IN TX BFR
005 136   271           CPB              /NO, IS CHAR = SRCH CHAR?
005 137   110 077 005   JFZ S1           /NO, CONT. SRCHING LINE
005 142   362           LLC              /SET PNTR TO INP BFR
005 143   106 137 001   CAL IN2          /INP CHANGES FROM KYBD
005 146   041           DCE              /BACK UP INP BFR PNTR
005 147   364           LLE              /SET PNTR TO LAST CHAR INP
005 150   307           LAM              /FETCH LAST CHAR INP
005 151   074 216       CPI 216          /CNT'L N?
005 153   150 230 005   JTZ NC           /YES, SRCH FOR NEW CHAR
005 156   074 224       CPI 224          /CNT'L T?
005 160   150 237 005   JTZ NX           /YES, SRCH FOR SAME CHAR
005 163   074 225       CPI 225          /CNT'L U?
005 165   150 257 005   JTZ SV           /YES, SAVE REST OF ORGIN. LINE
005 170   040           INE              /RESET INP BFR PNTR
005 171   106 147 002   S3, CAL CKOV     /CHECK FOR POSSIBLE BFR OVRFLO
005 174   106 221 002   CAL FND          /FETCH S.A. OF SRCH LINE
005 177   313           LBD              /SAVE S.A. IN REG B & C
005 200   324           LCE
005 201   106 204 002   CAL ZLOK         /FETCH S.A. OF NEXT LINE
005 204   056 000       LHI 000          /SET PG TO 000
005 206   106 331 003   CAL SDLT         /DELETE ORIGINAL LINE
005 211   066 173       LLI 173
005 213   327           LCM              /FETCH CHAR COUNT+1
005 214   106 224 004   CAL SNST         /INSERT LINE
005 217   066 162       LLI 162          /SET PNTR TO LINE NO.
005 221   307           LAM
005 222   106 344 002   CAL DECR         /DECR LINE NO. TO CORRECT FOR
                        /                 INCR IN SNST ROUTINE
005 225   104 033 001   JMP INCMD        /RET TO COMMAND MODE
                        /
005 230   106 000 006   NC, CAL RCV      /FETCH NEW SRCH CHAR
005 233   346           LEL              /SAVE INP BFR PNTR
005 234   066 144       LLI 144          /SAVE IN SRCH TBL
005 236   370           LMA
                        /
005 237   106 245 005   NX, CAL RT       /RESET REG FROM SRCH TBL
005 242   104 077 005   JMP S1           /CONT. SRCH
                        /
005 245   304           RT, LAE          /SAVE INP BFR PNTR
005 246   066 144       LLI 144
005 250   106 065 002   CAL LDBE         /LOAD REG FROM SRCH TBL
005 253   320           LCA              /SET INP BFR PNTR
005 254   104 316 002   JMP NCRD         /INCR TX BFR PNTR AND RET
                        /
005 257   106 245 005   SV, CAL RT       /RESET REG FROM SRCH TBL
005 262   364           S4, LLE          /SET TX BFR PNTR
005 263   353           LHD
```

```
005 264   307          LAM                 /FETCH NEXT CHAR FROM TX BFR
005 265   240          NDA                 /IS CHAR = EOL INDICATOR?
005 266   150 311 005  JTZ SCT             /YES, COMPLETE SRCH
005 271   020          INC
005 272   021          DCC                 /IS INP BFR FULL?
005 273   150 303 005  JTZ S5              /YES, IGNORE CHAR
005 276   362          LLC                 /NO, SET INP BFR PNTR
005 277   056 000      LHI 000
005 301   370          LMA                 /STORE CHAR IN INP BFR
005 302   020          INC                 /INCR INP BFR PNTR
005 303   106 316 002  S5, CAL NCRD        /INCR TX BFR PNTR
005 306   104 262 005  JMP S4              /CONT. STORING LINE
                       /
005 311   362          SCT, LLC            /SAVE INP BFR PNTR
005 312   106 256 001  CAL FBFLM           /FETCH INP BFR LIMIT
005 315   104 171 005  JMP S3              /INSERT NEW LINE IN TXT BFR
                       /


                       /
006 000                RCV,                /USER DEFINED INPUT ROUTINE
                                           /FOR OPERATOR'S INPUT DEVICE
                       /
006 100                READ,               /USER DEFINED INPUT ROUTINE
                                           /FOR BULK STORAGE DEVICE
                       /
006 200                PRINT,              /USER DEFINED OUTPUT ROUTINE
                                           /FOR OPERATOR'S DISPLAY DEVICE
                       /
006 300                PUNCH,              /USER DEFINED OUTPUT ROUTINE
                                           /FOR BULK STORAGE DEVICE
                       /
```

OPERATING THE EDITOR PROGRAM

THE READER WHO HAS STUDIED THE PRESENTATION OF THE VARIOUS ROUTINES THAT MAKE UP THE DESCRIBED EDITOR PROGRAM SHOULD HAVE LITTLE DIFFICULTY OPERATING THE PROGRAM. OPERATION CONSISTS OF SIMPLY INPUTTING COMMANDS WHEN THE EDITOR IS IN THE "COMMAND MODE" AND ENTERING THE DESIRED "TEXT" WHEN THE PROGRAM IS EXECUTING A COMMAND THAT EXPECTS TEXT TO BE PROCESSED!

IT IS ASSUMED THAT THE FIRST TIME THE PROGRAM IS LOADED INTO MEMORY THE USER WILL UTILIZE SOME SORT OF MANUAL SYSTEM. (HOPEFULLY SOMETHING ALONG THE LINES OF A KEYBOARD LOADER PROGRAM RATHER THAN TOGGLE SWITCHES!) IT GOES PRACTICALLY WITHOUT SAYING THAT ONE WOULD EXPECT THAT ALMOST THE FIRST THING ONE SHOULD DO WHEN THE PROGRAM HAS FIRST BEEN LOADED IS TO MAKE SOME SORT OF COPY OF THE OBJECT CODE ON A BULK STORAGE DEVICE SO THAT THE PROGRAM CAN BE EASILY LOADED INTO THE COMPUTER THEREAFTER! (THIS WOULD NATURALLY BE APPROPRIATE FOR THE USER PROVIDED I/O ROUTINES THAT WILL BE USED WITH THE EDITOR PROGRAM TOO!)

AS MENTIONED PRIOR TO THE PRESENTATION OF THE ASSEMBLED LISTING, THE PROGRAM IS STARTED BY DIRECTING THE COMPUTER TO JUMP TO LOCATION 033 ON PAGE 01 AND PLACING THE COMPUTER IN THE NORMAL PROGRAMMED OPERATION RUN MODE. ONCE THE PROGRAM HAS BEEN STARTED, DIRECTIVES TO THE PROGRAM ARE ISSUED VIA THE OPERATOR'S INPUT DEVICE. FOR THOSE THAT MAY HAVE SKIMMED

OVER THE PRESENTATIONS OF THE ROUTINES AND GENERAL INFORMATION PROVIDED IN THIS MANUAL, SOME EXAMPLES OF THE PROGRAM'S TYPICAL OPERATION WILL BE PRESENTED NEXT.

NOTE: IN THE FOLLOWING PRESENTATION THE ABBREVIATION C/R WILL BE USED FOR A CARRIAGE RETURN AND "CTRL/X" FOR A COMBINATION OF A "CONTROL CHARACTER" ENTRY ON THE OPERATOR'S INPUT DEVICE. OPERATOR INPUTS WILL BE UNDERLINED, PROGRAM RESPONSES WILL NOT BE UNDERLINED.

WHEN THE PROGRAM IS FIRST STARTED THE "COMMAND MODE" SYMBOL WILL BE DISPLAYED.

```
>
```

THE FIRST COMMAND GIVEN WHEN THE PROGRAM IS INITIALLY STARTED SHOULD BE A "KILL" COMMAND. THIS COMMAND EFFECTIVELY "CLEARS" THE MAIN TEXT BUFFER AND SETS UP THE UPPER BOUNDARY VALUE FOR THE TEXT BUFFER.

```
K(CTRL/L)

>
```

THE ">" ISSUED BY THE PROGRAM FOLLOWING THE "KILL" COMMAND INDICATES THE PROGRAM HAS EXECUTED THE COMMAND AND IS BACK IN THE COMMAND MODE AWAITANOTHER COMMAND. TO BEGIN PLACING TEXT INTO THE TEXT BUFFER (SUCH AS A SOURCE LISTING FOR A COMPUTER PROGRAM) ONE WOULD ISSUE THE "APPEND" COMMAND AND START ENTERING TEXT:

```
A(CTRL/L)
/THIS IS A SAMPLE(C/R)
/OF PROGRAM EDITING(C/R)
ORG 000 000(C/R)
START, INP 0(C/R)
LMA(C/R)
RST 0(C/R)
END(C/R)
```

WHEN THE OPERATOR WAS FINISHED ENTERING TEXT AND WANTED TO CHANGE FROM THE TEXT ENTERING MODE BACK TO THE COMMAND MODE, THE OPERATOR WOULD UTILIZE THE "CONTROL D" COMBINATION. THE PROGRAM WOULD RESPOND BY DISPLAYING:

```
>
```

TO EXAMINE THE CONTENTS OF THE TEXT BUFFER AT THIS TIME, THE OPERATOR COULD ISSUE THE "COMPLETE LIST" COMMAND BY ENTERING:

```
L(CTRL/L)
```

THE PROGRAM WOULD RESPOND BY DISPLAYING:

```
/THIS IS A SAMPLE
/OF PROGRAM EDITING
ORG 000 000
START, INP 0
LMA
RST 0
END

>
```

THAT IS, IT WOULD DISPLAY THE ENTIRE CONTENTS OF THE TEXT BUFFER AND THEN THE SYMBOL INDICATING IT HAD RETURNED BACK TO THE "COMMAND MODE."

IF, FOR EXAMPLE, THE OPERATOR DECIDED TO SAY, INSERT AN INSTRUCTION IN THE SOURCE LISTING BEING PREPARED WITH THE EDITOR PROGRAM, THEN ONE COULD USE THE "INSERT" COMMAND TO DO SOMETHING SUCH AS:

```
I 0 6(CTRL/L)
INL(C/R)
(CTRL/D)

>
```

THE ABOVE WOULD RESULT IN "INL" BECOMING LINE SIX IN THE EDITOR BUFFER. (NOTE WHEN SPECIFYING THE LINE NUMBER IN THE COMMAND HOW TWO NUMBERS MUST BE USED. HOWEVER, LEADING ZEROS IN A GROUP OF NUMBERS ARE NOT REQUIRED.) ONE MIGHT DESIRE TO VERIFY THE INSERTION OF THE NEW LINE BY REQUESTING A PARTIAL LISTING OF THE TEXT BUFFER:

```
L 0 5,0 10(CTRL/L)

LMA
INL
RST 0
END

>
```

SUPPOSE AT THIS TIME, THE OPERATOR DECIDED TO CHANGE THE "RST 0" INSTRUCTION IN THE LISTING TO "RST 2." ONE COULD USE THE "SEARCH" COMMAND TO IMPLEMENT THE CORRECTION:

```
S 0 7(CTRL/L)T
RST 2(C/R)

>
```

OR, THAT ONE DECIDE TO CHANGE THE "LMA" INSTRUCTION TO "LMB." ONE COULD USE THE "CHANGE" COMMAND AT THIS POINT.

```
C 0 5(CTRL/L)
LMB(C/R)
(CTRL/D)

>
```

THE OPERATOR THEN MIGHT ASK FOR A FINAL LISTING TO VERIFY THE REVISIONS:

```
L(CTRL/L)

/THIS IS A SAMPLE
/OF PROGRAM EDITING
ORG 000 000
START, INP 0
LMB
INL
RST 2
END

>
```

WITH THE SOURCE LISTING FINALLY PREPARED AS DESIRED, THE OPERATOR COULD THEN USE THE "WRITE" COMMAND TO PLACE THE CONTENTS OF THE TEXT BUFFER ON AN EXTERNAL STORAGE DEVICE FOR LATER USE (SAY BY AN ASSEMBLER PROGRAM) OR LATER RECALL BACK INTO THE EDITOR'S TEXT BUFFER.

THE EXAMPLES ON THE LAST SEVERAL PAGES DO NOT COVER ALL THE POSSIBLE OPERATIONAL CAPABILITIES OF THE EDITOR PROGRAM, BUT THEY DO SERVE AS A DEMONSTRATION OF THE TYPES OF OPERATIONS ONE MAY PERFORM. A QUICK REFERENCE SUMMARY OF THE VARIOUS OPERATIONS IS PROVIDED AS AN APPENDIX TO THIS MANUAL.

OTHER USES FOR AN EDITOR PROGRAM

BESIDES THE RATHER OBVIOUS USES OF USING AN EDITOR PROGRAM TO PREPARE TEXT (THIS MANUAL WAS PREPARED USING AN EDITOR PROGRAM) AND TO PREPARE SOURCE LISTINGS (THOSE SHOWN HEREIN WERE PREPARED USING AN EDITOR PROGRAM TOO), AN EDITOR PROGRAM CAN SERVE AS A VALUABLE "BASE" FOR A WIDE VARIETY OF PROGRAMS. IN FACT, IT CAN BE THE STARTING POINT FOR ALMOST ANY KIND OF THE PROGRAM WHERE A LARGE AMOUNT OF ALPHANUMERIC DATA NEEDS TO BE ENTERED INTO THE COMPUTER. ONCE THE INFORMATION IS IN THE MAIN TEXT BUFFER ONE MAY WRITE THE DATA OUT TO THE BULK STORAGE DEVICE AND HAVE IT PROCESSED BY ANOTHER PROGRAM (SUCH AS AN ASSEMBLER PROGRAM). OR, ONE CAN LEAVE THE DATA IN THE TEXT BUFFER AND LOAD IN ANOTHER PROGRAM (WITHOUT DISTURBING THE TEXT BUFFER AREA) THAT WILL PERFORM FURTHER OPERATIONS ON THE DATA IN THE TEXT BUFFER. OF COURSE, IF ONE HAS ENOUGH MEMORY IN A SYSTEM, ONE COULD EVEN HAVE AN ARRANGEMENT WHEREBY THE EDITOR PROGRAM, THE TEXT BUFFER AREA, AND ANOTHER OPERATING PROGRAM WERE ALL IN THE COMPUTER AT THE SAME TIME.

WHAT KINDS OF OTHER OPERATIONS COULD ONE PERFORM ON TEXT IN THE TEXT BUFFER? HERE ARE SOME IDEAS.

TEXT IN THE TEXT BUFFER COULD BE FURTHER PROCESSED BY A PROGRAM THAT WOULD "JUSTIFY" THE TEXT SO THAT EACH LINE WAS THE SAME LENGTH. THIS IS OFTEN DESIRABLE IF ONE WANTS TO PREPARE NEAT LOOKING REPORTS. THE PROCESS IS NOT TOO DIFFICULT. ONE DEVELOPS A FEW BASIC ALGORITHMS FOR PROCESSING EACH LINE OF TEXT WHICH DICTATE WHEN SPACES SHOULD BE INSERTED IN A LINE SO THAT EACH LINE ENDS AT THE SAME LOCATION. OF COURSE, THESE ALGORITHMS MAY REQUIRE THE OPERATOR TO FOLLOW A FEW RULES - SUCH AS MAKING SURE THAT THE INITIAL TEXT ENTRIES ARE WITHIN A CERTAIN RANGE. FOR INSTANCE, ONE MIGHT SET THE RULE THAT IF A LINE OF TEXT IS NOT BETWEEN 64 AND 72 CHARACTERS THAT NO ATTEMPT AT JUSTIFICATION IS TO BE MADE BY THE JUSTIFYING PROGRAM. (THIS WOULD STOP THE PROGRAM FROM ATTEMPTING TO JUSTIFY A PARTIAL LINE AT THE END OF A PARAGRAPH.) ONE COULD THEN PROCEED TO ESTABLISH SOME RULES FOR THE JUSTIFYING PROGRAM TO OPERATE UNDER SUCH AS THE FOLLOWING.

THE JUSTIFYING PROGRAM SHOULD NOT INSERT SPACES FOR THE PURPOSES OF JUSTIFYING A LINE PRIOR TO THE FIRST "PRINTING" CHARACTER. THIS IS DONE SO THAT THE BEGINNING LINE OF A PARAGRAPH MAY BE INDENTED BY THE OPERATOR WITHOUT THE JUSTIFYING PROGRAM LATER UPSETTING THE FIRST LINE FORMAT. THEN, ONE MIGHT SET UP RULES FOR SCANNING A LINE FOR PUNCTUATION MARKS. SUCH POINTS COULD BE USED AS THE PRIMARY PLACES AT WHICH TO HAVE THE JUSTIFYING PROGRAM INSERT SPACES IF NEEDED. (A GOOD RULE TO APPLY HERE IS TO ALWAYS TEST THE NEXT CHARACTER AFTER A PUNCTUATION MARK TO SEE IF IT CONTAINS A SPACE! IF NOT, THEN THE PROGRAM SHOULD NOT ATTEMPT TO JUSTIFY AT THAT POINT AS A "." MIGHT BE PART OF A GROUP OF DIGITS IN A NUMBER THAT CONTAINED A DECIMAL POINT - OR A PERIOD INSIDE A PARENTHESIS. (SUCH AS ILLUSTRATED IN THIS COMMENT.)

ONE MAY THEN DEVELOP RULES FOR THE JUSTIFYING PROGRAM TO FOLLOW IF VALID PUNCTUATION MARKS ARE FOUND IN A LINE. FOR INSTANCE, ONE COULD GO ALONG THE LINES OF DIVIDING ALL THE SPACES REQUIRED TO JUSTIFY THE LINE BETWEEN THE NUMBER OF PUNCTUATION MARKS IN THE LINE. FOR EXAMPLE, IF A LINE HAD TWO VALID PUNCTUATION MARKS (SUCH AS THE PREVIOUS LINE) AND ONE NEEDED TWO SPACES TO JUSTIFY THE LINE, ONE COULD HAVE THE PROGRAM INSERT ONE SPACE AFTER EACH PUNCTUATION MARK. FOR CASES WHERE THE DIVISION RESULTS IN ODD VALUES, ONE CAN ESTABLISH A PRIORITY RELATIONSHIP (SUCH AS A "." ALWAYS GETTING AN EXTRA SPACE OVER A COMMA, OR THE FIRST, OR MIDDLE, OR LAST PUNCTUATION MARK BEING THE EXTRA SPACE INSERTION POINT). IN CASES WHERE A LINE HAS NO PUNCTUATION MARKS BUT NEEDS JUSTIFICATION ONE CAN DICTATE PROGRAM RULES SUCH AS INSERTING SPACES AFTER EVERY ODD OR EVEN WORD. HOWEVER, SUCH A SIMPLE RULE CAN CREATE TEXT WITH A SKEWED APPEARANCE. A MORE SOPHISTICATED APPROACH IS TO STAGGER THE PATTERN SO THAT ONE SUCH LINE HAS EXTRA SPACES INSERTED AFTER ODD NUMBERED WORDS AND THE NEXT AFTER EVEN NUMBERED WORDS. OR TO ALTERNATELY INSERT SPACES NEAR THE MIDDLE OF A LINE, THEN TOWARDS ALTERNATING ENDS OF THE LINES.

BY NOW THE READER SHOULD BE ABLE TO UNDERSTAND THE CONCEPT OF DEVELOPING A JUSTIFYING PROGRAM. SUCH A PROGRAM COULD BE DEVELOPED AS AN OUTPUT ROUTINE OF THE EDITOR PROGRAM PRESENTED, OR AS AN ENTIRELY SEPARATE PROGRAM THAT MIGHT PROCESS TEXT FROM THE BULK STORAGE DEVICE. OR, AN ENTERPRISING PROGRAMMER JUST MIGHT TACKLE SUCH A PROJECT BY INCORPORATING IT INTO THE STRUCTURE OF THE EDITOR PROGRAM ITSELF (AS EACH LINE OF TEXT IS INPUTTED - AFTER ALL, EACH LINE OF TEXT IS RESIDING IN THE INPUT TEXT BUFFER WHICH PROVIDES A SUITABLE PLACE FOR FURTHER PROCESSING BEFORE IT IS TRANSFERRED TO THE MAIN TEXT BUFFER)! IN ANY EVENT, THE READER CAN OBSERVE THE LAST FEW JUSTIFIED PARAGRAPHS TO GET AN IDEA OF WHETHER SUCH CAPABILITY IS APPEALING TO THE INDIVIDUAL AND POSSIBLY WORTH WORKING TOWARDS!

AN EDITOR PROGRAM CAN BECOME THE INPUT PROGRAM FOR DEVELOPING A SYSTEM THAT USES A DATA BASE FOR SOME PURPOSE. FOR EXAMPLE, ONE COULD USE AN EDITOR TO CREATE LARGE FILES OF NAMES AND ADDRESSES. SUCH FILES MIGHT LATER BE PROCESSED BY ANOTHER PROGRAM FOR A VARIETY OF PURPOSES. WITH THE INTIMATE KNOWLEDGE OF HOW THE EDITOR PROGRAM OPERATES AFTER READING THIS PUBLICATION, ONE SHOULD BE IN A POSITION TO UNDERSTAND HOW A FEW SIMPLE TECHNIQUES CAN BE USED TO COUPLE THE BASIC CAPABILITIES IN WITH OTHER PROGRAMS.

SUPPOSE, FOR ILLUSTRATION, THAT ONE NEEDED TO DEVELOP A LARGE BASE OF NAMES AND ADDRESSES AND PERIODICALLY WANTED TO SAY, PREPARE A MAILING LIST COMPOSED OF SELECTED INDIVIDUALS. IT IS OBVIOUS THAT THE EDITOR PROGRAM MAY BE USED TO CREATE BLOCKS OF SUCH NAMES AND ADDRESSES BY SIMPLY ENTERING THEM INTO THE TEXT BUFFER AND THEN TRANSFERRING THEM TO THE SYSTEMS BULK STORAGE DEVICE SUCH AS A MAGNETIC TAPE UNIT. ANOTHER PROGRAM COULD THEN BE USED TO FURTHER PROCESS THE NAMES AND ADDRESSES. HOWEVER, WITH KNOWLEDGE OF HOW THE EDITOR PROGRAM WORKS, ONE CAN ARRANGE THINGS SO THAT THE PROGRAM THAT EVENTUALLY PROCESSES THE NAMES AND ADDRESSES, TO, FOR INSTANCE, PREPARE A MAILING LIST, CAN BE MADE SOMEWHAT LESS COMPLICATED.

ONE WAY TO APPROACH DEVELOPING SUCH A SYSTEM WOULD BE TO DEVELOP A "HEADER CODING" SYSTEM FOR EACH NAME AND ADDRESS. WITH SUCH A "HEADER" AN ENTRY MIGHT APPEAR AS SHOWN HERE:

```
061575 JRD1MAST12345 ABCD
JOHN R. DOE
123 MAIN ROAD
ANYTOWN, STATE  12345
```

WHERE THE TOP LINE IS A CODING SYSTEM THAT INCLUDES A DATE, A COMPOSITE CODE MADE UP OF THE INITIALS OF THE NAME, THE FIRST NUMBER AND LETTER OF THE STREET ADDRESS, THE FIRST LETTER OF THE CITY, INITIALS OF THE STATE, AND THE ZIP CODE OF THE ADDRESS, PLUS SOME ARBITRARY CODING.

ONE COULD SIMPLY ENTER A LARGE GROUP OF NAMES AND ADDRESSES IN THE FORMAT HYPOTHESIZED INTO THE TEXT BUFFER. WHEN THE TEXT BUFFER WAS FILLED ONE COULD OUTPUT THE DATA TO A BULK STORAGE DEVICE AND FILL UP THE TEXT BUFFER WITH A NEW BATCH. ANOTHER PROGRAM COULD THEN PROCESS THE DATA ON THE BULK STORAGE DEVICE.

HOWEVER, ONE COULD UTILIZE A TECHNIQUE THAT WOULD SIMPLIFY THINGS FOR THE PROGRAM THAT PROCESSED THE DATA. THE TECHNIQUE CONSIST OF SIMPLY SELECTING A COUPLE OF SPECIAL CHARACTERS TO MARK, FOR INSTANCE, THE BEGINNING AND END OF A "HEADER LINE." WHAT SPECIAL CHARACTERS? HOW ABOUT SOME "CONTROL" CHARACTERS THAT HAVE A CODE GREATER THAN 215 (OCTAL) AND THAT DO NOT CONFLICT WITH THE ONES USED IN THE "SEARCH" ROUTINE IN THE EDITOR PROGRAM? ONE MIGHT PICK "CONTROL P" (ASCII CODE 220) TO MARK THE START OF THE HEADER LINE AND "CONTROL Q" (ASCII CODE 221) TO INDICATE THE END OF THE HEADER LINE (WHICH IS THE START OF THE NAME AND ADDRESS SECTION). THESE PARTICULAR CODES WILL BE STORED BY THE EDITOR PROGRAM AS THOUGH THEY WERE TEXT CHARACTERS.

A PROGRAM TO PROCESS DATA STORED ON THE BULK STORAGE DEVICE COULD THEN EASILY DETECT THE START OF EACH "HEADER LINE." A HEADER LINE COULD THEN BE SCANNED TO DETERMINE IF THE NAME THAT FOLLOWED WAS IN A SELECTED GROUP THAT ONE WAS INTERESTED IN PROCESSING. AS SOON AS A "DECISION" IN THIS REGARDS HAD BEEN MADE, THE PROGRAM COULD EITHER QUICKLY ADVANCE TO THE END OF THE "HEADER LINE" AND OUTPUT THE DATA, OR PROCEED TO THE BEGINNING OF THE NEXT HEADER LINE.

ONE COULD ALSO EASILY PERFORM MANEUVERS SUCH AS COUNTING A CERTAIN NUMBER OF "HEADERS" TO ARRIVE AT A SELECTED POINT IN A FILE. THE READER MAY DISCERN OTHER TYPES OF USEFUL OPERATIONS USING THE CONCEPT PRESENTED.

THE TEXT EDITOR PROGRAM MAY BE USED AS AN INPUT PORTION OF A PROGRAM THAT SORTS OR ARRANGES DATA IN SOME DESIRED FASHION. THE TRICK IN USING IT FOR THIS PURPOSE IS TO USE TAB POINTS SO THAT ALL LINES TO BE SORTED OR OTHERWISE RE-ARRANGED BY ANOTHER PROGRAM ARE THE SAME LENGTH. (THIS MAKES THE DEVELOPMENT AND OPERATION OF THE SORT PROGRAM MUCH EASIER!) AS THE READER SHOULD NOW KNOW, THE ORIGINAL TAB VALUES SELECTED FOR THE EDITOR PROGRAM WERE AT INTERVALS OF EIGHT CHARACTERS. THIS IS A VERY CONVENIENT NUMBER (OR MULTIPLE THEREOF) TO USE IN A SORTING OPERATION. TO CLARIFY THE CONCEPT, SUPPOSE SOMEONE WANTED TO ARRANGE A LIST OF NAMES IN ALPHABETICAL ORDER AFTER THEY HAD BEEN ENTERED INTO A "FILE" IN RANDOM FASHION. TO USE THE EDITOR PROGRAM TO "SET UP" THE INITIAL DATA BASE, ONE WOULD PICK A TAB POSITION (SAY, THE 3'RD ONE - WHICH IS AFTER THE 24'TH CHARACTER IN A LINE) THAT WOULD ALLOW FOR THE LONGEST NAME TO BE ENTERED. ONE WOULD THEN SIMPLY ENTER THE NAMES USING THE EDITOR PROGRAM AND MAKING SURE THAT ONE "TABBED" OVER TO THE SAME POSITION AFTER EACH ENTRY AS ILLUSTRATED BELOW:

```
ZIEN, ROBERTO            *
DAVIS, THOMAS            *
WALTERS, ANTHONY         *
LISBON, PETRONOVICH      *
JONES, SALLY             *
LI, AL                   *
ANDERSON, ALLISON        *
PORTER, JANE             *
```

THE ASTERISK IN THE ILLUSTRATION INDICATES THE 3'RD TAB POINT AT WHICH EACH ENTRY WAS TERMINATED. WITH THE DATA IN THE TEXT BUFFER IN A FIXED LENGTH FORMAT, ONE CAN THEN USE A SIMPLE SORT PROGRAM TO RE-ARRANGE THE DATA, MOVING EACH ENTRY FROM ONE "LINE" TO THE NEXT AS REQUIRED. SINCE THE "LINES" ARE ALL THE SAME LENGTH THERE WILL BE NO NEED TO CONSTANTLY RESHUFFLE THE ENTIRE BUFFER AREA EACH TIME AN ENTRY IS MOVED IN FRONT OF OR IN BACK OF ANOTHER ENTRY. (A PERSON WHO REALLY WANTS TO DO LOT OF SUCH SORTING COULD MAKE THE SORT PROGRAM EVEN SIMPLER BY MODIFYING THE EDITOR PROGRAM SO THAT IT DID NOT PLACE A "ZERO BYTE" AT THE END OF EACH LINE - THEREBY MAKING ALL LINES AN EVEN MULTIPLE OF 8 (DECIMAL) LOCATIONS.)

THE USER WITH IMAGINATION (AND DESIRE) WILL PROBABLY FIND USES FOR AN EDITOR PROGRAM THAT ARE UNIQUE TO THE USER'S REQUIREMENTS. THE ABOVE SUGGESTIONS ARE INTENDED SOLELY TO STIMULATE THE MIND AND POINT OUT THAT AN EDITOR PROGRAM CAN OPEN THE DOOR TO A WHOLE NEW WORLD FOR THE SMALL SYSTEMS OWNER. IT IS HOPED THAT THE MATERIAL PRESENTED IN THIS PUBLICATION HAS AT LEAST CRACKED THE SEAL ON THAT DOOR!

APPENDIX - EDITOR COMMAND FORMAT SUMMARY

COMMAND	FORMAT	DESCRIPTION
APPEND	A	APPENDS TEXT TO END OF TEXT BUFFER
CHANGE	C XXX YYY	CHANGE CONTENTS OF LINE
	C XXX YYY,MMM NNN	OR LINES SPECIFIED
DELETE	D XXX YYY	DELETE LINE
	D XXX YYY,MMM NNN	OR LINES SPECIFIED
INSERT	I XXX YYY	INSERT LINE(S) BEFORE LINE SPECIFIED
LIST	L	LIST ENTIRE CONTENTS OF TEXT BUFFER
	L XXX YYY	LIST LINE SPECIFIED
	L XXX YYY,MMM NNN	LIST GROUP OF LINES SPECIFIED
READ	R	READ TEXT FROM BULK STORAGE DEVICE
WRITE	W	WRITE TEXT TO BULK STORAGE DEVICE
SEARCH	S XXX YYY *	SEARCH LINE SPECIFIED FOR CHARACTER (*) ENTERED IMMEDIATELY FOLLOWING COMMAND

SPECIAL "CONTROL" CHARACTERS

CHARACTER	FUNCTION
CONTROL D	ABORT CURRENT MODE AND RETURN TO COMMAND MODE
CONTROL I	PERFORM TAB SPACING FUNCTION
CONTROL L	EXECUTE COMMAND OR FUNCTION
CONTROL S	CLEAR CONTENTS OF TEXT INPUT BUFFER
CONTROL N	(SEARCH MODE) - CHANGE SEARCH CHARACTER TO NEXT CHARACTER ENTERED AND SEARCH LINE FOR APPEARANCE OF NEW SEARCH CHARACTER
CONTROL T	(SEARCH MODE) - CONTINUE SEARCHING FOR NEXT APPEARANCE OF SEARCH CHARACTER
CONTROL U	(SEARCH MODE) - APPEND REMAINDER OF ORIGINAL LINE TO REVISED LINE & PLACE IN TEXT BUFFER
RUBOUT	DELETE PREVIOUSLY ENTERED CHARACTER
CARRIAGE RETURN	TERMINATE CURRENT PROCESS OR LINE ENTRY