Tektronix®
COMMITTED TO EXCELLENCE

INFORMATION DISPLAY DIVISION PROGRAM EXCHANGE

TITLE	PART NUMBER
4051 Assembler	062-7456-01 - Program 11
ORIGINAL DATE: October 1983 / REVISION DATE:	EQUIPMENT, OPTIONS AND SOFTWARE REQUIRED (INCLUDING PERIPHERALS AND HOST SYSTEM): 4051 - 16K
AUTHOR: Carl Hovey & Steve Tuttle — Tektronix, Inc. Wilsonville, OR	

ABSTRACT

Files: 1 ASCII Program
1 ASCII Data
3 ASCII Text
(transfer to separate tape)

Statements: N/A (In Call Execute Format)

This program is an interactive assembly and debugging tool for the 4051. IT provides the 4051 programmer with a versatile system for creating and debugging programs in 6800 machine code. The one pass assembler allows the user to examine memory locations and registers.

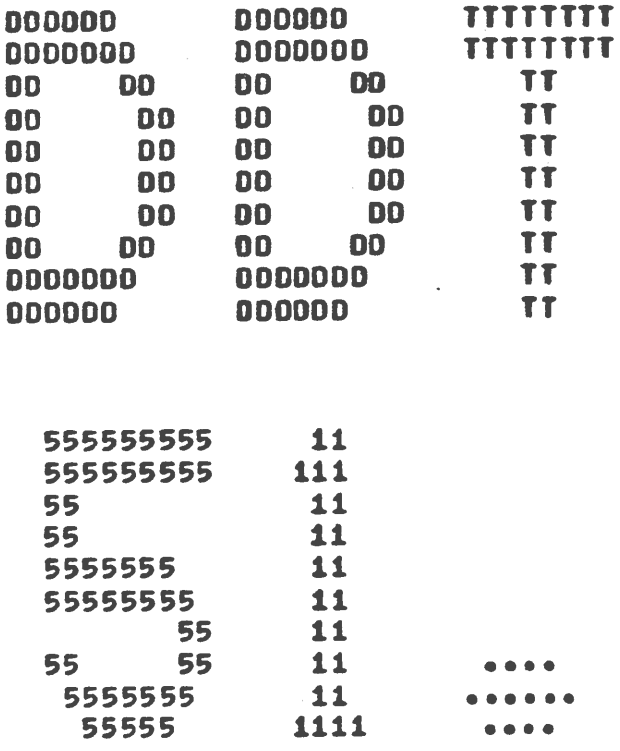
To create source files the Editor ROM or any other text processing program for the 4051 can be used. Also source can be entered directly into the program.

PRELIMINARY OPERATING INSTRUCTIONS

The five files comprising this program must be transferred to a separate tape.

Using the instructions on page ii, transfer the following:

From TEKNIQUES VOL. 7 NO. 4 T2 tape			TO 4051 Assembler tape		
File #	Bytes	Type	File #	Bytes	Type
40	768	ASCII Program	1	768	ASCII Program
41	11520	ASCII Data	2	11520	ASCII Data
42	4608	ASCII Text	3	4608	ASCII Text
43	3328	ASCII Text	4	3328	ASCII Text
44	768	ASCII Text	5	768	ASCII Text



DDT V51.7 IS AN INTERACTIVE DISSASSEMBLY AND DEBUGGING TOOL FOR THE 4051. ITS PURPOSE IS TO PROVIDE THE 4051 SYSTEMS PROGRAMMER AND ROMPACK WRITER WITH A VERSATILE SYSTEM FOR CREATING AND DEBUGGING HIS 6800/4051 CODE. DDT V51.7 CONSISTS OF TWO PROGRAMS ON 4051 MAG TAPE

DEBUGGER EXPLAINATION

THE 4051 RESIDENT DEBUGGER IS LOADED INTO MEMORY BY A SYSTEM INVOLVING THREE CHARACTER STRINGS. ONE CHARACTER STRING CONTAINS THE DEBUGGER IN A RELOCATABLE ASCII HEX FORMAT. THE SECOND STRING IS A RELOCATING LOADER. THE LOADER IS IN CALL "EXEC" FORMAT. THE THIRD STRING IS THE "DESTINATION" STRING. IN OTHER WORDS THE LOADER LOADS THE DEBUGGER INTO THE THIRD STRING. THE LOADER BUILDS THE THIRD STRING FROM THE FIRST STRING BY TAKING THE ASCII HEX CHARACTERS OF THE FIRST STRING AND BUILDING EIGHT BIT MACHINE CODE BYTES AND STORING THEM IN THE THIRD STRING. CHARACTERS OF A VALID CHARACTER STRING MAY **NOT** HAVE THE EIGHT BIT SET SO THE THIRD CHARACTER STRING OF THIS SETUP IS INVALID. HOWEVER, THERE IS A TRICK. THE THIRD STRING IS DIMENSIONED LARGE ENOUGH TO HOLD THE BINARY VERSION OF THE DEBUGGER BUT WHEN THE LOADER BUILDS THE THIRD STRING IT DOES NOT CHANGE THE LENGTH VALUE FOR THAT STRING. THUS THE 4051 DOES NOT REALIZE THERE IS A STRING WHICH IS FULL OF INVALID CHARACTERS.

THE USER MAY SPECIFY THE ABSOLUTE MEMORY ADDRESS INTO WHICH THE DEBUGGER IS TO BE LOADED BY DEFINING THE THIRD STRING (DESTINATION STRING) TO CONTAIN THE ABSOLUTE HEX ADDRESS FOR THE BEGINNING OF THE DEBUGGER. IF A LOAD ADDRESS IS NOT SPECIFIED BY THE USER, THE DEBUGGER IS LOADED INTO THE SPACE ALLOCATED BY THE 4051 FOR THE THIRD STRING. A USER MUST BE VERY CAREFUL ABOUT WHERE HE LOADS THE DEBUGGER BECAUSE IT IS POSSIBLE TO LOAD IT RIGHT OVER TOP OF PERTINENT DATA ALREADY IN RAM. IF SOME RAM IS CLOBBERED, NOTHING WILL GO WRONG UNTIL THE USER TRIES TO RETURN TO BASIC. ON A RETURN TO BASIC WITH CLOBBERED RAM A SYSTEM ERROR IS ENCOUNTED AND **ALL** DATA IN MEMORY IS WIPED OUT.

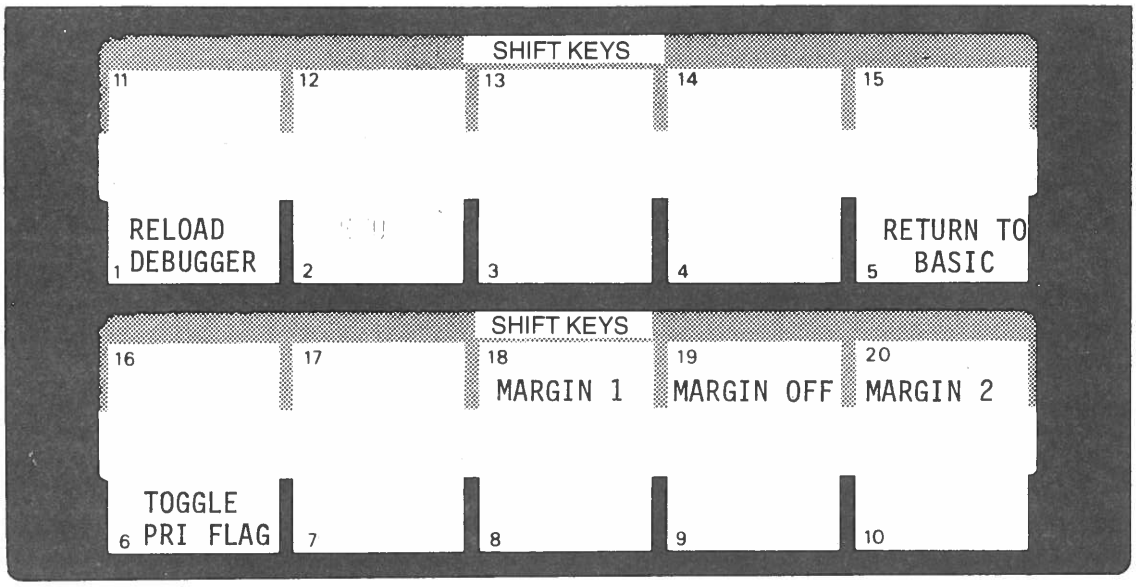
IF NO RAM IS CLOBBERED WHEN THE DEBUGGER IS LOADED IT IS **POSSIBLE** TO RUN THE DEBUGGER AND RETURN TO BASIC, RUN BASIC AND THEN RETURN TO THE DEBUGGER ANY NUMBER OF TIMES, WITH NO ILL EFFECTS ON EITHER THE USER OR THE SYSTEM.

A NOTE OF CAUTION: WHEN YOU ARE RUNNING THE DEBUGGER YOU HAVE ACCESS TO THE VERY GUTS OF THE 4051 OPERATING SYSTEM AND IT IS VERY, VERY EASY TO STEP ON SOMETHING AND THEN THE WHOLE SYSTEM WILL CRASH.

TITLE DDT V51.7

TAPE #

FILE #



DDT V51.7 USER DEFINABLE KEYS

KEY #1: RELOAD DEBUGGER PROGRAM.
THIS KEY WILL RELOAD THE DEBUGGER PROGRAM'S MACHINE CODE FROM A SOURCE STRING INTO A DESTINATION STRING; CONTROL IS THEN TRANSFERRED TO THE DEBUGGER. THIS KEY FUNCTION ONLY WORKS AFTER THE DEBUGGER'S LOADER PROGRAM HAS DEFINED ALL OF THE NECESSARY STRINGS.

KEY #5: RETURN TO BASIC.
THIS KEY WILL TRANSFER CONTROL BACK TO THE 4051'S BASIC OPERATING SYSTEM. ALL SYSTEM VARIABLES CLOBBERED BY THE DEBUGGER ARE RESTORED. SYSTEM VARIABLES ALTERED BY THE USER MAY RESULT IN A FATAL SYSTEM ERROR. THIS KEY FUNCTION WILL NOT WORK IF CONTROL WAS TRANSFERRED TO THE DEBUGGER PROGRAM VIA A BREAK POINT (USE G COMMAND INSTEAD).

KEY #6: TOGGLE PRINT FLAG.
THIS KEY TOGGLES A FLAG WHICH WILL SUPPRESS THE AUTOMATIC ECHO ON A DEPOSIT INSTRUCTION, DEPOSIT HEX, OR DEPOSIT ASCII COMMAND. MOSTLY USED TO SPEED UP THE ASSEMBLER. ON ENTRY TO THE DEBUGGER, THIS FLAG DEFAULTS TO 'ECHO ON'.

THE FOLLOWING THREE USER DEFINABLE KEYS CONTROL THE CRT DISPLAY'S MARGIN MODES.

KEY #18: MARGIN 1.
THIS KEY WILL FORCE THE DEBUGGER PROGRAM TO BLINK THE "F" AND WAIT FOR THE USER TO PAGE THE SCREEN AFTER ONE COLUMN OF DATA HAS BEEN PRINTED.

KEY #19: MARGIN OFF.
THIS KEY WILL FORCE THE DEBUGGER TO IGNORE ALL PAGE FULL CONDITIONS. AFTER TWO COLUMNS HAVE BEEN PRINTED THE DEBUGGER WILL WRITE OVER COLUMN ONE.

KEY #20: MARGIN 2.
THIS KEY WILL FORCE THE DEBUGGER PROGRAM TO BLINK THE "F" AND WAIT FOR THE USER TO PAGE THE SCREEN AFTER TWO COLUMNS OF DATA HAVE BEEN PRINTED. THE DEBUGGER DEFAULTS TO THIS MODE

NOTE: THE DATA COMMUNICATIONS OVERLAY MAY HELP TO REMEMBER THE KEY NUMBERS

RESIDENT ASSEMBLER

LABELS

ALL LABELS BEGIN WITH THE LETTER "Z".

LABELS MAY END WITH A <CARRIAGE RETURN>, A <PERIOD> OR A <SPACE>. IF THE LABEL ENDS WITH A SPACE, A 6800 OP CODE MNEMONIC, OR AN "*" WITH A COMMENT SHOULD FOLLOW.

EXAMPLE:

```
INPUT NEXT VALID CHARACTER, IGNORE CHARACTERS IN ZTABLE
ZGETCHR                    * EQUATE THE LABEL GETCHR TO THE VALUE $57C9
$6000                      * ORG $6000
ZSTART   LDX    #ZTABLE    * LOAD POINTER TO TABLE
ZANOTHER JSR     ZGETCHR   * REGISTER A GETS NEXT CHARACTER
         JSR     ZSCAN     * SCAN FOR A MATCH
         BEQ     ZANOTHER  * IF MATCH, GET ANOTHER CHARACTER
         RTS
ZSCAN    CMP A   0,X
         BEQ     ZDONE     * IF A MATCHES THEN RETURN WITH EQUAL
         INX
         TST     0,X       * MSB SET MEANS END OF TABLE
         BPL     ZSCAN
ZDONE    RTS
$7000                      * ORG TABLE AT $7000
ZTABLE                     * CHARACTERS TO IGNORE
"$07."$19."$53."$61."$69."$72."$FF.
```

LABELS MAY BE OF ANY LENGTH AND MAY CONTAIN ANY CHARACTER EXCEPT FOR A <PERIOD>, <CARRIAGE RETURN>, OR <SPACE>.

THERE IS A WAY TO CLEAR THE SYMBOL TABLE: RELOADING THE DEBUGGER IS PROBABLY THE EASIEST WAY. THERE ARE NO CHECKS FOR SYMBOL TABLE OVERFLOW.

COMMENTS

COMMENTS MAY APPEAR IN THE SOURCE CODE AS SHOWN IN THE EXAMPLE ABOVE. COMMENTS BEGIN WITH "*" AND END WITH <CARRIAGE-RETURN>.

$HEX

IN ORDER TO EASE THE TRANSFER OF SOURCE CODE FROM THE DEBUGGER UP TO THE CYBER WITH A MINIMUM OF TRANSLATION, THE DEBUGGER WILL ALLOW A "$" BEFORE HEXIDECIMAL NUMERALS.

UNRESOLVED REFERENCES

IF ONE EXECUTES A JSR TO A LABEL THAT IS NEVER DEFINED, THE STACK WILL GROW UNTIL THE SYSTEM BLOWS UP. SO MAKE SURE ALL LABELS ARE DEFINED BEFORE RUNNING A PROGRAM. THIS PROBLEM IS A SIDE EFFECT OF THE WAY THE ONE PASS ASSEMELER HANDLES FOWARD REFERENCES.

"WHAT" ERROR MESSAGES

THE "WHAT" ERROR MESSAGE GIVES THE FOLLOWING INFORMATION:
ADDRESS IN LINE BUFFER OF THE CHARACTER CAUSING THE ERROR
ASCII CHARACTER CAUSING THE ERROR
HEXIDECIMAL EQUIVALENT OF THE ASCII CHARACTER CAUSING THE ERROR

EXAMPLE
0224 V 56 WHAT?

MACROS ON TAPE

ONE MAY SAVE A SEQUENCE OF DEBUGGER COMMANDS ON TAPE. THE SEQUENCE MAY CONTAIN ANY DEBUGGER COMMAND, BUT SOME OF THEM MAY GET YOU INTO TROUBLE (LIKE "J"). THE INTENT IS TO ALLOW THE ASSEMBLY OF A FILE OF SOURCE CODE, OR TO ALLOW ONE TO EXECUTE A COMMON SEQUENCE OF COMMANDS. EXECUTION OF THE MACRO ENDS WITH AN END OF FILE MARK ($FF) AS PLACED ON THE
TAPE BY A CLOSE STATEMENT IN BASIC. IF TWO BACK-TO-BACK CARRIAGE-RETURNS ARE ENCOUNTERED, EXECUTION OF THE MACRO IS SUSPENDED BUT MAY BE RESUMED LATER.

TO EXECUTE THE MACRO, FIND THE FILE (SEE THE "H" COMMAND) AND HIT RIGHT BRACE <SHIFT-]> (THE KEY NEXT TO <BACK SPACE>) AND RETURN.

USING INTERRUPTS
COMMAND SYNTAX NOTATION

COMMAND SYNTAX NOTATION

NOTATION CONVENTIONS USED IN COMMAND SPECIFICATIONS AND EXAMPLES THROUGHOUT THIS MANUAL ARE LISTED BELOW.

NOTATION	DESCRIPTION
BOLDFACE CHARACTERS	BOLDFACE CHARACTERS MUST BE ENTERED EXACTLY AS SHOWN.
ORDINARY CHARACTERS	ORDINARY CHARACTERS (I.E., NOT BOLDFACE) IDENTIFY ELEMENTS THAT MUST BE REPLACED WITH USER-SELECTED VALUES.
$	WHEN NOT USED IN THE CONTEXT OF A DEBUGGER COMMAND, THE DOLLAR SIGN INDICATES A HEXADECIMAL STRING. THE LEGAL DIGITS ARE 0-9 AND A-F.
< >	A WORD OR WORDS ENCLOSED IN THE SYMBOL PAIR <> STANDS FOR AN ENTITY THAT CANNOT BE PRINTED DIRECTLY, OR WHICH IS OTHERWISE REPRESENTABLE ONLY BY A PHRASE. (I.E., THE SYMBOL **<CR>** REPLACES THE PHRASE "CARRIAGE RETURN," AND STANDS FOR THE ASCII CHARACTER WHOSE HEXADECIMAL VALUE IS $0D.)
[	THIS IS HOW A SQUARE BRACKET LOOKS ON THIS PRINTER
[ ]	AN ELEMENT ENCLOSED IN ORDINARY BRACKETS (NOT BOLDFACE) IS OPTIONAL. IF BRACKETS ARE NESTED, ELEMENTS IN INNER LEVELS MAY NOT BE SPECIFIED UNLESS THOSE IN THE OUTER LEVELS ARE ALSO SPECIFIED. FOR EXAMPLE, THE SPECIFICATION [<ITEM #1>[<ITEM #2>]], MEANS THAT THE ONLY LEGAL CHOICES ARE: (1) NEITHER ITEM; (2) ITEM #1 ONLY; OR, (3) ITEMS #1 AND #2. ITEM #2 MAY NOT BE SELECTED ALONE.
<CONTROL-I>	THE CHARACTER PRODUCED BY HOLDING THE "CONTROL" KEY DOWN WHILE TYPING AN "I" (ASCII $09).
<CONTROL-B>	THE CHARACTER PRODUCED BY HOLDING THE "CONTROL" KEY DOWN WHILE TYPING A "B" (ASCII $02).
<CONTROL-Y>	THE CHARACTER PRODUCED BY HOLDING THE "CONTROL" KEY DOWN WHILE TYPING A "Y" (ASCII $19).

<CR>	THE CARRIAGE RETURN (ASCII $0D).
<ESC>	THE CHARACTER PRODUCED BY THE "ESCAPE" KEY ON THE TERMINAL (ASCII $1B).
<LF>	THE LINE FEED (ASCII $0A).
<RUBOUT>	THE RUBOUT CHARACTER (ASCII $7F).

DDT V51.7 OPERATIONS

STRINGING COMMANDS

COMMANDS MAY BE STRUNG TOGETHER ON A LINE, BY THEMSELVES OR INSIDE A REPEAT LOOP. ANY COMMAND THAT MAY BE TERMINATED BY A PERIOD (', ", AND M COMMANDS) MUST BE THUS TERMINATED UNLESS IT IS THE LAST COMMAND ON A LINE.

REGAINING CONTROL

REGAINING CONTROL

DDT V51.7 HAS A COMPACT COMMAND NOTATION, AND WILL INTERPRET WHAT YOU TYPE DISCONCERTINGLY LITERALLY. THUS, IT IS SOMETIMES NECESSARY TO RECOVER FROM PROBLEMS THAT TYPING ERRORS OR HASTY ENTRIES HAVE CAUSED. THERE ARE TWO METHODS FOR ASSERTING CONTROL, LISTED IN ORDER OF ASCENDING POWER AND DESCENDING COUTH.

BREAK KEY

HIT THE GOLD BREAK KEY TWICE AND HOPE YOU GET BACK TO BASIC. ALSO HOPE THAT THE CONTENTS OF MEMORY ARE STILL INTACT.

THE LAST RESORT

TURN OFF THE POWER. (CONDOLENCES)

DEBUGGER INPUT CONTROL

INPUT LINE TERMINATION

<CR>

THE CARRIAGE RETURN (**<CR>**) IS THE UNIVERSAL TRAILING DELIMITER AND LINE TERMINATOR. ANY INPUT STRING, LEGAL OR ILLEGAL, MAY BE TERMINATED BY A **<CR>** (THOUGH NOT NECESSARILY WITHOUT ERROR).

THE **<CR>** IS A LEGAL TERMINATOR FOR THE * COMMAND (INSTRUCTION ENTRY) AND THE " COMMAND (HEXADECIMAL DIGIT ENTRY). THE **<CR>** WILL CORRECTLY TERMINATE THESE COMMANDS ONLY AT THE POINTS WITHIN THE COMMAND SYNTAX AT WHICH A TERMINATOR IS LEGAL. AN ERROR MESSAGE (DEPENDENT UPON THE COMMAND INVOLVED) WILL BE GENERATED IF THE **<CR>** IS USED IN OTHER PLACES.

<LF>

THE LINE FEED (**<LF>**) MAY **NOT** BE SUBSTITUTED FOR THE CARRIAGE RETURN (**<CR>**) FOR THIS VERSION OF DDT.

CORRECTING TYPING ERRORS

TO CORRECT TYPING ERRORS USE THE 4051 LINE EDITOR KEYS AS DESCRIBED IN THE 4051 MANUAL.

CAUTION: THE LINE EDITOR KEYS CLOBBER THE 4051 PSEUDO REGISTER R0.

DDT V51.7 OPERATIONS
REPEATING DEBUGGER COMMANDS

REPEATING DEBUGGER COMMANDS

MANY TIMES IT IS DESIRABLE TO PERFORM AN OPERATION REPETITIVELY, AS WHEN ONE WISHES TO INITIALIZE MEMORY TO A SPECIFIC VALUE. TWO COMMANDS ARE PROVIDED THAT ALLOW COMMANDS OR COMMAND SEQUENCES TO BE EXECUTED REPETITIVELY.

REPEAT LOOP DEFINITION

[M];N[

THE [COMMAND SETS UP THE PARAMETERS FOR A REPEAT LOOP. THE FORMAT MAY BE ;N[OR M;N[, WHERE N IS THE REPEAT COUNT AND M IS AN OPTIONAL STARTING ADDRESS FOR THE OPERATIONS CONTAINED IN THE REPEAT LOOP. THE MAXIMUM VALUE FOR N IS $FFFF (DECIMAL 65,535). NESTING OF REPEAT LOOPS IS NOT PERMITTED. REPEAT LOOPS ARE TERMINATED BY THE] COMMAND, OR BY THE END OF THE LINE. WHEN THE END OF THE LINE IS THE TERMINATOR FOR A REPEAT LOOP, THE LOOP IS EXECUTED ONLY ONCE. THIS IS NOT CONSIDERED AN ERROR.

REPEAT LOOP TERMINATION

]

THE] COMMAND MARKS THE END OF A REPEAT LOOP. NO PARAMETERS ARE ALLOWED. A [COMMAND NOT TERMINATED BY A] COMMAND IS NOT TREATED AS AN ERROR, BUT EXECUTES ONLY ONCE. A] COMMAND NOT PRECEDED BY A [COMMAND IS IGNORED.

MEMORY DISPLAY, MODIFICATION, AND SEARCH

DISPLAYING MEMORY IN INSTRUCTION FORMAT

[M][;[N]]/

MEMORY DUMP (INVOKED BY / AND IMPLIED BY >, <, AND ^), USES THE CURRENT ADDRESS (CURADR) AS A POINTER TO THE NEXT "INSTRUCTION." IF THE BYTE CAN BE DECODED TO AN INSTRUCTION MNEMONIC, THIS ROUTINE PRINTS THE BYTE(S) IN HEX, THE MNEMONIC AND THE OPERAND. THE LENGTH LOCATION (LEN) IS SET TO THE NUMBER OF BYTES DUMPED.

IF M IS PRESENT, IT IS SUBSTITUTED FOR THE CURRENT ADDRESS BEFORE THE LOCATION IS DUMPED. IF THE SEMICOLON IS PRESENT (OR IF M IS PRESENT) AUTOMATIC ADDITION OF THE LENGTH OF THE PREVIOUS OPERATION IS SUPPRESSED. OTHERWISE, THE LENGTH FROM THE PREVIOUS STEP IS ADDED TO THE CURRENT ADDRESS TO DETERMINE THE LOCATION TO BE DISPLAYED. IF N IS NOT PRESENT, THE ASSUMPTION IS THAT ONE LOCATION IS TO BE DUMPED. IF N IS PRESENT, MORE THAN ONE LOCATION MAY BE DUMPED BY A SINGLE COMMAND. N IS TREATED MODULO 256, AND A VALUE OF ZERO IN THE LOWER 8 BITS CAUSES 256 "INSTRUCTIONS" TO BE DUMPED.

THE DUMP FORMAT IS SIMILAR TO AN ASSEMBLY LISTING. THE GENERAL FORMAT IS:

ADDR/ XX MNE R ARG,X <E.A.>

FIELD	MEANING
ADDR	ADDRESS OF THE FIRST BYTE OF THE "INSTRUCTION"
/	INDICATES THE DUMP WAS CAUSED BY A / COMMAND
XX	FIRST BYTE OF THE "INSTRUCTION"
MNE	THE INSTRUCTION MNEMONIC. PRESENT ONLY IF THE FIRST BYTE IS A LEGAL MC6800 INSTRUCTION OP CODE.
R	REGISTER DESIGNATOR. PRESENT ONLY FOR THOSE INSTRUCTIONS THAT REQUIRE IT.
ARG	ARGUMENT FIELD ENTRY. PRESENT ONLY FOR THOSE INSTRUCTIONS THAT REQUIRE AN ARGUMENT. THE ARGUMENT ENTRY MAY BE IMMEDIATE (SIGNALED BY A "#" PRECEEDING THE NUMBER), OR IT MAY BE AN ADDRESS. IN EITHER CASE, THE NUMBER MAY BE ONE OR TWO BYTES LONG.
,X	INDEXED OPERATION INDICATOR. PRESENT ONLY FOR INDEXED INSTRUCTIONS.
<E.A.>	EFFECTIVE ADDRESS. PRESENT FOR INDEXED AND BRANCH INSTRUCTIONS ONLY.

DISPLAYING THE REFERENCED ADDRESS

[M]>

THE '>' COMMAND ALLOWS ONE TO EXAMINE A LOCATION REFERENCED BY THE PREVIOUS INSTRUCTION EXAMINED. ONE MAY THEN RETURN TO THE ORIGINAL INSTRUCTION WITH THE < COMMAND. ALL THIS HOPPING AROUND IN MEMORY MAY BE DONE WITHOUT HAVING TO REMEMBER OF RETYPE ANY OF THE ADDRESSES AS THEY ARE SAVED FOR YOU BY THE DEBUGGER. ISN'T THAT NICE? IF M IS PRESENT, IT IS SUBSTITUTED FOR THE CURRENT ADDRESS BEFORE THE LOCATION IS DUMPED.

THE FORMAT OF THE DUMP IS SIMILAR TO THAT OF THE / COMMAND.

FORWARD STEPS ARE SAVED IN THE DEBUGGER'S STACK FOR LATER DISPLAY BY THE < COMMAND. BE CAREFUL NOT TO OVERFLOW THE STACK.

BACKTRACKING

<

THE < COMMAND BACKS UP TO THE INSTRUCTION EXAMINED BEFORE THE LAST > COMMAND. IF THERE ARE NO PREVIOUS REFERENCES TO BACK UP TO, THE ERROR MESSAGE < **WHAT?** IS OUTPUT.

DISPLAYING THE PRECEEDING ADDRESS

[M]^

THE ^ ("CARET" OR "UP ARROW") COMMAND EXAMINES THE CONTENTS OF THE PREVIOUS LOCATION. THE ^ COMMAND STEPS BACKWARD ONE BYTE AND DISPLAYS THE LOCATION. IF M IS PRESENT, IT IS SUBSTITUTED FOR THE CURRENT ADDRESS BEFORE THE LOCATION IS DUMPED.

MEMORY DUMP IN HEX AND ASCII (QUICK DUMP)

[M][;[N]]Q

THE Q COMMAND DUMPS MEMORY IN BOTH HEX AND ASCII. THE COMMAND FORMAT IS M;NQ, WHERE M AND N ARE OPTIONAL, AND THE SEMICOLON IS NECESSARY ONLY IF N IS SPECIFIED. IF M IS PRESENT, IT IS SUBSTITUTED FOR THE CURRENT ADDRESS BEFORE THE LOCATION IS DUMPED.

THE DUMP BEGINS AT THE CURRENT ADDRESS AND INCREMENTS THE CURRENT ADDRESS AS IT GOES, LEAVING IT SET ONE BYTE BEYOND THE LAST LOCATION DUMPED. THE NUMBER OF LINES TO DUMP IS SPECIFIED BY N (EACH LINE CONSISTS OF 16 BYTES). IF N IS NOT PRESENT, IT IS ASSUMED EQUAL TO 1. N IS TREATED MODULO 256, AND A ZERO IN THE LOW BYTE WILL CAUSE 256 LINES TO BE DUMPED.

THE FORMAT OF EACH LINE OF THE DUMP IS THE ADDRESS OF THE FIRST BYTE ON THE LINE (4 HEX DIGITS), FOLLOWED BY 8 GROUPS OF TWO BYTES (2 DIGITS PER BYTE, 4 DIGITS PER GROUP), FOLLOWED BY THE ASCII TRANSLATIONS OF THE BYTES. THE MOST SIGNIFICANT BIT IS IGNORED AND NO CHARACTERS ARE CONVERTED TO A PRINTABLE CHARACTER WITH AN UNDERSCORE.

PRINTING A FLOATING POINT VALUE

[M][;N] U

THE U COMMAND CONVERTS A 4051 FLOATING POINT NUMBERS FROM THE INTERNAL BINARY FORMAT TO ASCII DECIMAL FORMAT AND PRINTS THE VALUES. M POINTS TO THE FIRST BYTE OF AN 8 BYTE FLOATING POINT NUMBER.

ENTERING MC6800 INSTRUCTIONS

[M][;]'<INSTRUCTION>

THE ' COMMAND ALLOWS THE ENTRY OF INSTRUCTIONS IN MNEMONIC FORM. A <SPACE> MAY BE SUBSTITUTED FOR THE "'" IN ORDER TO MAKE THE INPUT LOOK MORE LIKE THE CYBER ASSEMBLER FORMAT. THERE ARE CERTAIN EXCEPTIONS TO THE ASSEMBLER INPUT FORMAT, THE MOST IMPORTANT BEING THAT ALL NUMBERS ARE ASSUMED TO BE HEXADECIMAL. A "$" IS PERMITTED BEFORE NUMBERS, BUT IS NOT REQUIRED. ALL LABELS MUST BEGIN WITH THE LETTER "Z". EACH INSTRUCTION MAY BE TERMINATED BY A <CR>, A SPACE (OR SPACES) WITH AN "*" FOLLOWED BY A COMMENT, OR A PERIOD (.). WHEN TERMINATED BY A PERIOD, ADDITIONAL DEBUGGER COMMANDS (INCLUDING INSTRUCTION ENTRIES) MAY BE PLACED ON THE SAME LINE.

THE SEMICOLON HAS NO EFFECT IF ENTERED WITH A VALUE FOR M. WHEN AN INSTRUCTION IS ENTERED WITH ONLY A SEMICOLON PRECEEDING THE ', AUTOMATIC ADDITION OF THE LENGTH OF THE PREVIOUS OPERATION IS SUPPRESSED. THIS ALLOWS (FOR EXAMPLE) AN INSTRUCTION TO BE DUMPED BY THE / COMMAND, AND THEN REPLACED BY ANOTHER INSTRUCTION WITHOUT THE NECESSITY OF TYPING THE ADDRESS AGAIN.

ENTERING HEXADECIMAL STRINGS

[M][;]"[XX]...

THE " COMMAND ALLOWS THE ENTRY OF UP TO THREE BYTES OF HEXADECIMAL DATA INTO MEMORY. M AND THE SEMICOLON ARE OPTIONAL AND THE XX ARE HEXADECIMAL DIGIT PAIRS (UP TO THREE PAIRS, AS INDICATED BY THE ...), WHICH MAY NOT BE SEPARATED BY SPACES. THE DIGITS OF A HEXADECIMAL DIGIT PAIR, WHICH CONSTITUTES ONE BYTE TO BE STORED INTO MEMORY, MAY NOT BE SEPARATED BY SPACES. A HEXADECIMAL STRING MAY BE TERMINATED BY A <CR>, OR A PERIOD (.). WHEN TERMINATED BY A PERIOD, ADDITIONAL DEBUGGER COMMANDS (INCLUDING HEXADECIMAL STRINGS) MAY BE PLACED ON THE SAME LINE.

IF M IS PRESENT, THE CURRENT ADDRESS IS REPLACED BY THE VALUE OF M BEFORE ANY BYTES ARE STORED. IF THE SEMICOLON IS PRESENT (OR IF M IS PRESENT) THE LENGTH FROM THE PREVIOUS OPERATION IS IGNORED. IF NEITHER M NOR THE SEMICOLON ARE PRESENT, THE LENGTH FROM THE PREVIOUS OPERATION IS ADDED TO THE CURRENT ADDRESS BEFORE ANY BYTES ARE STORED.

AFTERPWHEN A TERMINATOR IS ENCOUNTERED AFTER ONE OR MORE BYTES HAVE BEEN STORED, THE ROUTINE EXITS WITH THE CURRENT ADDRESS SET TO THE ADDRESS OF THE FIRST BYTE STORED, AND THE LENGTH SET TO THE NUMBER OF BYTES STORED.

ENTERING ASCII STRINGS INTO MEMORY

[M]<AMPERSAND>[<ASCII CHARACTERS>]

THE <AMPERSAND> TELLS THE DEBUGGER TO ENTER ASCII CHARACTERS INTO MEMORY. THIS COMMAND TERMINATES UPON A <CR>. THE <CR> IS NOT ENTERED INTO MEMORY: TO ENTER A <CR> INTO MEMORY, DEPOSITE AN HEXADECIMAL 0D.

ODT V51.7 OPERATIONS
MEMORY DISPLAY, MODIFICATION, AND SEARCH

MOVING BLOCKS OF MEMORY

S;EMD

THE M COMMAND ALLOWS BLOCKS OF DATA TO BE MOVED ABOUT IN MEMORY. THE S PARAMETER IS THE STARTING ADDRESS OF THE BLOCK TO BE MOVED, E IS THE END OF THE BLOCK TO BE MOVED (INCLUSIVE), AND D IS THE DESTINATION ADDRESS. ALL THREE ADDRESSES MUST BE PRESENT. EACH M COMMAND MAY BE TERMINATED BY A **<CR>**, OR A PERIOD (.). WHEN TERMINATED BY A PERIOD, ADDITIONAL DEBUGGER COMMANDS (INCLUDING ADDITIONAL M COMMANDS) MAY BE PLACED ON THE SAME LINE.

IF ANY OF THE ADDRESSES ARE MISSING, THE START ADDRESS FOLLOWS THE END ADDRESS, OR THE DESTINATION BLOCK WOULD WRAP OVER THE END OF MEMORY, NO BYTES ARE MOVED AND AN ERROR MESSAGE IS PRINTED.

A NOTE OF CAUTION IS WORTHWHILE AT THIS POINT. IF AN NMI INTERRUPT OCCURS WHILE THE M COMMAND IS EXECUTING, THE RESULTS WILL BE DISASTROUS. THE STACK POINTER IS USED AS ONE OF THE INDICES DURING EXECUTION OF THIS COMMAND, SO AN INTERRUPT WILL LIKELY AS NOT DESTROY THE DATA THAT IS TO BE MOVED.

SEARCHING MEMORY

[M];NL

THE L (LOOK) COMMAND SEARCHES MEMORY FROM ONE ADDRESS TO ANOTHER (HIGHER) ADDRESS FOR A SPECIFIED BIT PATTERN (IN THE "ONES" REGISTER) UNDER A MASK (IN THE "MASK" REGISTER). THE LENGTH OF THE SEARCH OBJECT MAY BE ONE, TWO, OR THREE BYTES (IMPLIED BY THE CONTENTS OF "MASK"). M IS OPTIONAL AND WILL DEFAULT TO THE CURRENT ADDRESS. M AND N ARE THE STARTING AND ENDING ADDRESSES, RESPECTIVELY. IF A MATCH IS FOUND, THE ADDRESS OF THE THREE-BYTE BLOCK THAT INCLUDES THE MATCHING LOCATIONS IS PRINTED IN THE FORMAT F XXXX (WHERE XXXX IS THE HEXADECIMAL ADDRESS). IF NO MATCH IS FOUND, THE DEBUGGER PROMPTS FOR THE NEXT COMMAND. ADDITIONAL L COMMANDS WITH NO ARGUMENTS WILL SEARCH FURTHER.

SEE REGISTER CHANGE OPERATIONS FOR SETTING THE "ONES" AND "MASK" REGISTERS.

REGISTER DISPLAY AND MODIFICATION

DISPLAYING ALL PROCESSOR REGISTERS

P

THE P COMMAND PRINTS THE CONTENTS OF THE USER'S REGISTERS IN THE FOLLOWING ORDER:

REGISTER	MNEMONIC
CONDITION CODES	C
ACCUMULATOR B	B
ACCUMULATOR A	A
INDEX REGISTER	X
PROGRAM COUNTER	P
STACK POINTER	S

NO PARAMETERS ARE LEGAL FOR THE "P" COMMAND.

SETTING REGISTERS

M=R
$[N]=R (ALTERNATE FORM)

THE = COMMAND CHANGES THE CONTENTS OF ANY ONE OF THE USER'S REGISTERS OR THE DEBUGGER REGISTERS MASK AND ONES. N IS OPTIONAL AND ASSUMED EQUAL TO ZERO IF NOT PRESENT, AND R IS A MNEMONIC WHICH DESIGNATES THE REGISTER TO BE SET. THE MNEMONICS ARE:

REGISTER	MNEMONIC
ACCUMULATOR A	A
ACCUMULATOR B	B
CONDITION CODES	C
DEBUGGER'S MASK REGISTER	M
DEBUGGER'S ONES REGISTER	O
PROGRAM COUNTER	P
STACK POINTER	S
INDEX REGISTER	X

DISPLAYING SINGLE REGISTERS

RR

AN "R" FOLLOWED BY A REGISTER MNEMONIC WILL DISPLAY THE CONTENTS OF THAT REGISTER.

EXECUTING SUBROUTINES

[M] J

THE J COMMAND TRANSFERS CONTROL FROM THE DEBUGGER TO THE SUBROUTINE STARTING AT M, OR IF M IS NOT SPECIFIED, THE CONTENTS OF REGISTER P IS USED. THE SUBROUTINE MAY RETURN CONTROL TO THE DUBUGGER WITH AN RTS. BREAKPOINTS ARE NOT ENABLED.

UPON RETURN TO THE DEBUGGER, ALL THE 6800'S REGISTERS EXCEPT FOR THE PROGRAM COUNTER ARE SAVED FOR EXAMINATION. (SEE DISPLAYING REGISTERS.)

TO ALLOW THE BREAK KEY TO WORK, THE J COMMAND CLEARS THE USER INTERRUPT MASK BEFORE PASSING CONTROL TO THE SUBROUTINE. THUS IF THE SUBROUTINE HANGS, A BREAK-BREAK MIGHT GET YOU BACK TO BASIC.

MAG TAPE OPERATIONS

MAG TAPE OPERATIONS

THE 4051 MAG TAPE UNIT MAY BE USED BY THE RESIDENT DEBUGGER. FILES MAY BE FOUND, READ AND WRITTEN. FILES MUST BE PRE-MARKED BY BASIC. NOTE: FILES WRITTEN BY THE DEBUGGER MIGHT NOT BE READ BY THE 4051 BASIC O.S. UNLESS GREAT CARE IS TAKEN TO FOLLOW BASIC'S FILE STRUCTURE, ESPECIALLY WITH REGARD TO THE BLOCK LENGTH. AND FILES WRITTEN BY BASIC MAY BE READ BY THE DEBUGGER ONLY ONE RECORD PER T COMMAND.

THERE ARE FIVE DEBUGGER MAG TAPE COMMANDS.

H FIND HEADER OF TAPE FILE

W WRITE DATA ON TAPE

T READ DATA FROM TAPE

- BACK UP ONE RECORD (256 BYTES)

N BACK UP TO BEGINNING OF FILE, TO ALLOW REWRITING HEADERS.

THE SYNTAX OF EACH COMMAND IS AS FOLLOWS:

1. NH FIND THE NTH FILE ON THE TAPE AND READ THE HEADER INTO THE STANDARD BASIC TAPE BUFFER. THE N H COMMAND IS EQUIVALENT TO THE FIND N STATEMENT IN BASIC, EXCEPT THAT **N IS HEXIDECIMAL. BE CAREFUL.**

2. XXXX;XXXXW XXXX REPRESENT A HEX NUMBER. THE FIRST HEX NUMBER IS THE MEMORY ADDRESS AT WHICH TO START READING DATA AND THE SECOND HEX NUMBER IS THE MEMORY ADDRESS AT WHICH TO STOP READING DATA. IN OTHER WORDS, DATA IN AN AREA OF 4051 RAM DELIMITED BY THE TWO HEX ADDRESSES IS WRITTEN AS ONE CONTIGUOUS DATA BLOCK ON TO THE 4051 MAG TAPE. THE TAPE MUST BE PRE MARKED BY BASIC.

IF XXXX;XXXX IS NOT SPECIFIED, THE STANDARD TAPE BUFFER FROM $11F TO $21E IS USED.

3. XXXX;XXXXT

DATA IS READ FROM MAG TAPE AND LOADED INTO 4051 RAM STARTING AT THE HEX ADDRESS SPECIFIED IN THE COMMAND. NOTE: THE TAPE MUST BE POSITIONED AT THE BEGINNING OF A VALID DATA BLOCK. THE SIZE OF THE DATA (THE DIFFERENCE BETWEEN STARTING ADDRESS AND ENDING ADDRESS) SHOULD MATCH THAT WHICH WAS WRITTEN.

IF THE XXXX;XXXX IS NOT SPECIFIED, THE STANDARD TAPE BUFFER FROM $11F TO $21E IS USED.

HOST LINK OPERATIONS

HOST LINK OPERATIONS

HOST LINK/DOWNLOADER

K

THE K COMMAND ALLOWS THE USER TO LINK TO A HOST TIME-SHARING SYSTEM AND TO DOWNLOAD ASCII HEX INTO THE 4051'S MEMORY. THE FORMAT OF THE COMMAND IS K. NO PARAMETERS ARE VALID.

BEFORE USING THE K LINK IN THE DEBUGGER THE 4051 ENVIRONMENTAL PARAMETER "RATE" MUST BE SET. "RATE" IS THE BAUD RATE AT WHICH YOU WISH TO COMMUNICATE WITH A HOST COMPUTER.FOR EXAMPLE:

```
CALL "RATE",1200,0,2
```

REFER TO THE 4051 OPT 1 MANUAL FOR FURTHER DETAILS. ONCE THE RATE IS SET, CONTROL MAY BE PASSED TO THE DEBUGGER AND THE K LINK MAY BE USED. THE RATE PARAMETER MUST BE SET AFTER A SYSTEM RESET OR POWER UP. TO ENTER THE K LINK FROM THE DEBUGGER TYPE <K> FOLLOWED BY A CARRIAGE RETURN. IT IS NOW POSSIBLE TO COMMUNICATE WITH AND DOWNLOAD FROM THE HOST IN THE STANDARD FASHION. TO EXIT THE K LINK HIT USER KEY #5. THIS GETS YOU BACK INTO THE DEBUGGER COMMAND MODE.

COMMAND SUMMARY

REGAINING CONTROL

USER KEY #5	RETURN TO BASIC

DEBUGGER INPUT CONTROL

<CR>	TERMINATE INPUT LINE
<RUBOUT>	DELETE LAST CHARACTER OF CURRENT INPUT LINE

(SEE 4051 MANUAL FOR A DESCRIPTION OF THE LINE EDITOR KEYS)

REPEATING DEBUGGER COMMANDS

[M];N[	REPEAT LOOP DEFINITION
]	MARK THE END OF A REPEAT LOOP

MEMORY DISPLAY, MODIFICATION, AND SEARCH

[M][;[N]]/	DISPLAY MEMORY IN INSTRUCTION FORMAT
[M]>	DISPLAY REFERENCED ADDRESS IN INSTRUCTION FORMAT
<	BACKTRACK FROM A PREVIOUS > COMMAND, DISPLAYING THE REFERENCING INSTRUCTION IN INSTRUCTION FORMAT
[M]^	DISPLAY PRECEDING ADDRESS IN INSTRUCTION FORMAT
[M][;[N]]Q	DISPLAY MEMORY IN HEXADECIMAL FORMAT WITH ASCII TRANSLATION (QUICK DUMP)
[M][;N]U	CONVERT MEMORY FROM FLOATING POINT TO ASCII

[M][;]'<INSTRUCTION>	ENTER INSTRUCTIONS INTO MEMORY
[M][;]"[XX]...	ENTER HEXADECIMAL DIGITS INTO MEMORY
<AMPERSAND>[<ASCII>]	ENTER ASCII CHARACTERS INTO MEMORY
S;EMD	MOVE BLOCKS OF MEMORY
[[M];N]L	SEARCH MEMORY FOR THE STRING SPECIFIED IN THE O REGISTER, UNDER THE MASK IN THE M REGISTER

REGISTER DISPLAY AND MODIFICATION

RR	DISPLAY SINGLE REGISTER
P	DISPLAY USER REGISTERS
;[N]=R	SET REGISTER

HOST LINK OPERATIONS

K	ENTER KRONOS LINK/DOWNLOADER

MAG TAPE OPERATIONS

NH	FIND FILE N
[XXXX;XXXX]W	WRITE DATA ONTO TAPE
[XXXX;XXXX]T	READ DATA FROM TAPE
-	BACK UP ONE RECORD
N	BACK UP TO BEGINNING OF FILE

This document presents an overview of the I/O System. It also contains a detailed description of the commonly used I/O System variables.

THE 4051 I/O SYSTEM WAS DESIGNED TO BE AS MODULAR AS POSSIBLE TO PROVIDE FOR AS MUCH DEVICE INDEPENDENCE AS RESONABLE.

THE FOLLOWING DISCRIBES THOSE I/O VARIABLES REFERED TO AS THE NORMAL I/O SYSTEM VARIABLES IN THE ROUTINE DOCUMENTATION. THESE VARIABLES MUST BE SET UP BEFORE CALLING MANY OF THE I/O ROUTINES. SOME ROUTINES DO EXIST FOR MANAGEMENT OF MANY OF THESE VARIABLES, THESE ROUTINES INCLUDE ADRDEV, BFRALC, UNADR.

NOTE: ^HNN IS THE NOTATION USED TO REPPESENT NN AS A HEX (BASE 16) NUMBER.

```
************** IOFUNC          ; PRESENT I/O OPERATION (BASIC KEYWORD)
************** A.STAT          ; CHANNEL A STATUS
;       STATUS BYTE FORMAT
DIRCT  = ^H80                  ; DIRECTION 0-OUTPUT 1-INPUT
BFRSTT = ^H20                  ; BUFFER STATUS 0-EMPTY 1-FULL
BUSACT = ^H10                  ; ACTIVE BUS 0-NO 1-YES
FMTVLD = ^H08                  ; SET IF USING STATEMENT APPLIES
ATVLD  = ^H04                  ; SET IF <ATSIGN> OR <PERCENTSIGN>
                               ; INFORMATION IS ON STACK
SECDEF = ^H02                  ; SECONDARY DEFINED 0-NO 1-YES
PRIDEF = ^H01                  ; PRIMARY DEFINED 0-NO 1-YES
;
************** A.PRIM          ; PRESENT PRIMARY ADDRESS
************** A.SEC           ; PRESENT SECONDARY ADDRES
************** A.STRT          ; POINTER TO FIRST VALID
                               ; CHARACTER POSITION IN THE BUFFER.
************** A.MAX           ; POINTER TO LAST VALID CHARACTER
                               ; POSTITION IN THE BUFFER.
************** A.PTR           ; MOVING POINTER IN I/O BUFFER
                               ; IT GENERALLY POINTS TO THE NEXT
                               ; USEABLE CHARACTER SLOT.
************** A.END           ; MOVING POINTER IN I/O BUFFER
                               ; DURING OUTPUT IT POINTS TO THE
                               ; MOST RECENTLY OUTPUT CHARACTER.
************** A.MAX           ; POINTER TO LAST VALID CHARACTER
                               ; POSTITION IN THE BUFFER.
;
;
;
************** PPMODE          ; <PERCENTSIGN> MODE FLAG
                               ; (SET IF <PERCENTSIGN> ENVOKED)
************** IECRLF          ; CR VS. CR/LF FLAG
************** NOOUT           ; OUTPUT BUFFER FLAGS
NOWRIT = ^H01                  ; IF SET NEVER OUTPUT BUFFER
LSTFMT = ^H02                  ; SET TO OUTPUT IN LIST MODE
SNDIT  = ^H80                  ; SET TO PUT EOI WITH LAST BYTE IN BUFFER
```

The following variables are of importance during input operations.

```
************** CRSTAT          ; INPUT BUFFER STATUS BYTE
                                ; (CRSTAT=0 IF NO DELIMITER IN BUFFER)
       CRNORM  = ^H01           ; NORMAL EOLCHR <CR> FOUND
       CRDC3   = ^H02           ; NOT USED
       CREOI   = ^H04           ; EOI FOUND
       CREOF   = ^H08           ; EOF FOUND (FILE ONLY)
       CRETX   = ^H10           ; ETX FOUND (LOGICAL END-OF-FILE CHAR.)
       CREOT   = ^H20           ; EOT FOUND (EOF ON INTERNAL TAPE)
       EOFTYP  = ^H38           ; (EOF+ETX+EOT)
       CRVLD   = ^H80           ; CRSTAT IS VALID IF SET
                                ; (REACHED END OF BUFFER)
************** EOLCHR          ; CHARACTER USED AS EOL DELIMITER
                                ; NOTE: THIS CHARACTER IS ALSO
                                ; USED DURING OUTPUT FUNCTIONS!!!!
************** ETXCHR          ; CHARACTER USED AS EOF DELIMITER
************** NULCHR          ; CHARACTER USED AS NULL CHARACTER
                                ; NOTE: IF 128. OR GREATER NO CHARACTER
                                ; WILL BE USED AS A NULL CHARACTER.
```

4051 SYMBOL DEFINITIONS

CONTAINED WITHIN IS A LIST OF THE 4051 SYSTEM GLOBAL SYMBOLS. THE SYMBOLS ARE LISTED IN ALPHABETICAL ORDER. EACH SYMBOL ENTRY CONTAINS INFORMATION IN THE FOLLOWING FORMAT:

```
SYMBOL(XXXX)T    COMMENT ABOUT THE SYMBOL.

        SYMBOL  SPECIFIES THE GLOBAL SYMBOL NAME.
        (XXXX)  SPECIFIES THE VALUE ASSOCIATED WITH THE GLOBAL SYMBOL.
        T       SPECIFIES THE TYPE OF THE SYMBOL ENCODED AS BELOW:
                   E -- SYMBOL REFERS TO AN ENTRY POINT FOR AN INDIVIDUAL
                        ROUTINE.  THE ADDRESS OF THE ENTRY POINT IS INDICATED
                        BY THE (XXXX) ENTRY.
                   C -- SYMBOL IS A CONSTANT WITH THE VALUE XXXX.
                   V -- SYMBOL IS A GLOBAL VARIABLE WITH THE ADDRESS XXXX.
                        NOTE: IF XXXX IS LESS THAN 100 (HEX) THEN THE
                        VARIABLE IS ON PAGE ZERO AND MAY BE ADDRESS USING THE
                        DIRECT MODE.
                   M -- SYMBOL IS A MAJOR MODULE NAME (THE NAME ASSOCIATED
                        WITH EACH BLOCK OF LISTINGS).  THE VALUE XXXX
                        ASSOCIATED WITH THIS SYMBOL IS MEANINGLESS (AT LEAST
                        PSEUDO RANDOM).
```

A Symbols

A0X	(AA43)E	ANX - ROUTINES TO ADD N TO THE X REGISTER.
A5X	(AA3E)E	" "
A6X	(AA3D)E	" "
A7X	(AA3C)E	" "
A8X	(AA3B)E	" "
A9X	(AA3A)E	" "
A10X	(AA39)E	" "
A11X	(AA38)E	" "
A12X	(AA37)E	" "
A13X	(AA36)E	" "
A14X	(AA35)E	" "
A15X	(AA34)E	" "
A16X	(AA33)E	" "
ABS	(EE06)E	ROUTINE TO GET ABSOLUTE FLOATING POINT VALUE.
ABSCOD	(003F)C	CODE FOR ABS TOKEN.
ACCEL	(05A4)V	KEYBOARD DRIVER ACCELERATION RATE.
ACIA	(87C6)C	THE ADDRESS OF THE ACIA IN THE COMM BACK PACK.
ACS	(00A8)C	BANK SWITCH CONTROL CONSTANT.
ACSCOD	(0048)C	TOKEN CODE FOR ARC COS FUNCTION.
ADBANK	(95DE)M	THE PART OF THE MATH PACK THAT IS IN THE OVERFLOW BANK.
ADMAIN	(E87A)M	THE PART OF THE ADVANCED MATH PACK IN THE MAIN BANK.
ADRDEV	(AE22)E	ROUTINE TO SET UP I/O SYSTEM STATUS INCLUDES A.PRIM, A.SE SET UP.
AFPITT	(B05D)E	ROUTINE TO CONVERT ASCII TO FPN, INTERNAL BUFFERS.
ALLCOD	(00AE)C	CODE FOR ALL TOKEN.
ALLTG	(0014)C	ALL TAG - STACK TAG FOR DELETE ALL CONSTRUCT.
ANCRNT	(0066)V	AUTO NUMBER CURRENT - THE CURRENT VALUE FOR AUTO NUMBER TO OUTPUT.

AND (EE25)E SUBROUTINE TO DO LOGICAL AND OF OPERANDS ON STACK.

ANDCOD(000A)C CODE FOR AND TOKEN.

ANHOLD(0462)V AUTO NUMBER HOLD - IF NOT ZERO DO NOT AUTO NUMBER ON THE NEXT LINE.

ANINCR(0068)V AUTO NUMBER INCREMENT - THE CURRENT INCREMENT FOR AUTO NUMBER.

ANYINT(050B)V INTERRUPTS PENDING COUNT FOR ROM PACK RECALL WHEN SYSTEM IS SAFE.

APPCOD(0059)C CODE FOR APP TOKEN.

APPEND(9204)E THE APPEND STATEMENT ROUTINE.

APPOLD(D9EC)E ROUTINE TO PERFORM I/O PORTION OF APPEND.

AS (AF82)E ROUTINE TO PROCESS AS. STACKS A TAG FOR USE LATER.

AS2COD(0051)C CODE FOR AS2 TOKEN.

A.END (0095)V MOVING POINTER IN I/O BUFFER. DURING OUTPUT IT POINTS TO THE MOST RECENTLY OUTPUT CHARATERS.

A.MAX (0093)V POINTER TO LAST VALID CHARACTER POSITION IN THE BUFFER.

A.PRIM(0090)V PRESENT PRIMERY ADDRESS.

A.PTR (0098)V MOVING POINTER IN I/O BUFFERR. IT GENERALLY POINTS TO THE NEXT USABLE CHARACTER SLOT.

A.SEC (0097)V PRESENT SECONDARY ADDRESS.

A.STAT(008F)V I/O SYSTEM STATUS FLAG BYTE.

A.STRT(0091)V POINTER TO FIRST VALID CHARACTER POSITION IN THE BUFFER.

ASC (0058)C BANK SWITCH CONTROL CONSTANT.

ASCCOD(001E)C CODE FOR ASC TOKEN.

ASCFPN(B55F)E CONVERTS STRING OF ASCII CHARACTERS INTO A FLOATING POINT NUMBER.

ASCOD (0035)C CODE FOR AS TOKEN.

ASG (C003)E ASSIGN - THE ROUTINE TO IMPLEMENT NORMAL ASSIGNMENT.

ASGCOD(008C)C CODE FOR ASG TOKEN.

ASN (00A0)C BANK SWITCH CONTROL CONSTANT.

ASNCOD(0047)C CODE FOR ARC SIN TOKEN.

ASSCOD(0074)C CODE FOR ASS TOKEN.

ASSCSC(C030)E ASSIGN A SCALER TO A SCALER - SYSTEM UTILITY.

ASSIGN(100C)C ASSIGN COMMAND I/O SYSTEM CONTROL CONSTANT.

ATLOAD(830D)E AUTO LOAD FUNCTION - CALLED FROM IDLE LOOP IF AUTO LOAD IS REQUIRED.

ATN (00B0)C BANK SWITCH CONTROL CONSTANT.

ATNCOD(0049)C TOKEN CODE FOR ARC TAN.

ATNOFF(F588)E ROUTINE TO TURN OF ATN GPIB LINE.

ATNON (F57F)E ROUTINE TO TURN ON ATN GPIB LINE.

ATPROC(AF94)E ROUTINE TO PROCESS @<ATSIGN> OR <PERCENTSIGN> INFORMATION ON THE STACK.
SETS UP A.PRIM AND A.SEC.

ATSNTG(0012)C STACK TAG TO MARK @<ATSIGN> OR <PERCENTSIGN> INFORMATION.

AUTONO(8279)E AUTO NUMBER - CALLED FROM THE IDLE LOOP TO OUTPUT LINE NUMBERS IF REQUIRED.

AXICOD(006D)C CODE FOR AXI TOKEN.

AXIS (000C)C BANK SWITCH CONTROL CONSTANT.

B Symbols

BACKUP(011D)E ROUTINE TO BACK UP ONE STACK ENTRY, R0 IS FAKE SP.

BAKARS(FE81)E MAGTAPE DRIVER ROUTINE TO BACKUP ONE PHYSICAL RECORD.

BAKSTG(0013)C STACK TAG FOR I/O SYSTEM BACKWARD POINTERS INTO THE I/O LIST ON THE STACK.

BANK (007C)V THE SAVED VALUE THAT THE BANK SWITCH IS CURRENTLY SET TO.

BANKSW(87C0)C THE ADDRESS OF THE BANK SWITCH REGISTER.

BELCAL(C727)E ROUTINE TO BEEP THE BELL.

BFRALC(AEC7)E ROUTINE TO ALLOCATE I/O BUFFERS BASED UPON A.PRIM.

BINCTL(CC56)M BINARY I/O CONTROL MODULE.

BLINK (00AE)V CURSOR BLINK CONTROL. SET TO DO WRITE-THRU CHARACTERS.

BNKADR(8800)C BANK ADDRESS - THE ADDRESS OF THE BANK SWITCHED AREA.

BNKEND(8816)C BANK END - THE ADDRESS OF THE END OF THE HEADER FIELD IN A BANK.

BNKMAP(8800)M BANK MAP - THE MODULE WITH THE HEADER FOR THE OVERFLOW ROM.

BOF (00JA)C MAGTAPE DRIVER BEGGINING OF FILE FLAG.

BPIAMT(003A)V BUFFER FOR PIAMTB.

BRKCNT(043D)V BRACKET COUNT FOR DIMENSION/SUBSCRIPT (DIMSUB) ROUTINE.

BRKCOD(003C)C CODE FOR BRK TOKEN.

BSTMT (C0AC)M BASIC STATEMENT PROCESSORS.

BYTCAL(E5C4)V PERFORMS THE FOLLOWING CALCULATION ((AB+(INT1*INT2))*8)+5.

BYTCNT(001E)V BYTE ALLOCATION COUNT.

C Symbols

CALCOD(006A)C CODE FOR CAL TOKEN.

CALL (E286)E ROUTINE TO IMPLEMENT THE CALL STATEMENT.

CALLBS(E286)M MODULE CONTAINING CALL STATEMENT PROCESSOR.

CALLTG(0017)C CALL TAG - STACK TAG FOR CALL ENTRY.

CASCOD(009C)C CODE FOR CAS TOKEN.

CASE (C248)E ROUTINE TO IMPLEMENT THE SET CASE STATEMENT.

CAT (0088)C BANK SWITCH CONTROL CONSTANT.

CATCOD(0017)C CODE FOR CAT TOKEN.

CCODES(0098)V BUFFER FOR CONDITION CODES IN MAGTAPE DRIVER.

CDOPTR(043B)V CURRENT DATA OBJECT POINTER - THE ADDRESS OF THE CURRENT DATA ITEM IN A DATA STATEMENT.

CDSPTR(0439)V CURRENT DATA STATEMENT POINTER - THE ADDRESS OF THE CURRENT DATA STATEMENT. IF ZERO FIND THE FIRST DATA STATEMENT IN THE PROGRAM.

CHAR (008D)V GLOBAL CONTAINING FIRST CHARACTER OF ASCII STRING.

CHR (006C)C BANK SWITCH CONTROL CONSTANT.

CHRCNT(0088)V ACTUAL CHARACTER COUNT DURING STRING INPUT OR READ.

CHRCOD(001C)C CODE FOR CHR TOKEN.

CHRVCT(0473)V VECTOR TO OPTIONAL CHARACTER PAINTER ROUTINE.

CHSCAN(C888)E ROUTINE TO PAINT CHARACTERS ON THE DISPLAY. THIS IS THE ACTUAL PAINTER NOT THE ENTIRE CONTROLLER.

CIDLE (C65C)E ROUTINE THAT DISPLAYS THE CURSOR WHILE WAITING FOR KEYBOARD I/O TO COME TO A LOGICAL END.

CLOCOD(007E)C CODE FOR CLO TOKEN.

CLOSE (F7AA)E ROUTINE TO HANDLE THE CLOSE COMMAND.

CLPTR (004D)V CURRENT LINE POINTER - THE ADDRESS OF THE STATEMENT BEING EXECUTED.

CLRARG(CE74)E CLEAR ARGUMENT - SPECIAL ENTRY POINT INTO TYPARG.

CLRBNK(A9D4)E CLEAR BANK - ROUTINE TO SET THE BANK SWITCH TO ZERO.

CMD (1A0C)C COMMAND COMMAND I/O SYSTEM CONTROL CONSTANT.

CMDCOD(0083)C CODE FOR CMD TOKEN.

CMDTBL(BB1A)E COMMAND TABLE - A SECTION OF THE EVALUATOR TABLES.

CMPFPN(B7C8)E ROUTINE TO COMPARE TWO FLOATING POINT NUMBERS: ONLY ONE IS ON STACK.

CNFRCS(FC57)E MAGTAPE DRIVE ROUTINE TO CLEAR NFR COUNTER.

CNT (0031)V FLAG WHICH CONTROLS FLOATING POINT INPUT CONVERSION.

CNTL (0591)V KEYBOARD DRIVER CONTROL KEY FLAG.

COL (0596)V KEYBOARD COLUMN FLAG.

COLCNT(009E)V I/O SYSTEM MATRIX COLUMN COUNT.

COLCOD(008D)C CODE FOR COL TOKEN.

COLCT (00AC)V TEMP. COUNTER FOR DISPLAY CHARACTER PAINTER (COLUMN).

COMCNT(0609)V COMMA REMAINDER COUNT.

COMCOD(009C)C CODE FOR COM TOKEN.

COMFLG(00F5)V COMMA FORMAT FLAG.

COMP (0C3A)V COMPRESS LINE FLAG FOR UNCOMPILE.

COMPR (F9AA)E THE MEMORY COMPRESS (GARBAGE COLLECTION) ROUTINE.

CONCOD(00B6)C CODE FOR CON TOKEN.

CONV1 (050C)V POINTER TO CONVERSION FACTOR FOR CURRENT TRIG MODE.

COPCOD(0073)C CODE FOR COP TOKEN.

COPTYP(0A2C)C COPY COMMAND I/O SYSTEM CONTROL CONSTANT.

COPY (F7C8)E ROUTINE TO HANDLE THE COPY COMMAND.

COSCOD(0042)C CODE FOR COS TOKEN.

COSTHA(04E2)V COSINE OF ROTATED ANGLE FOR RELATIVE GRAPHICS.
IT'S AN FPN.

CPSIEC(C49D)E ROUTINE TO SET CR VS. CR/LF. PFI@37,26:1 <OR 0>

CPSSET(C479)E ROUTINE TO SET OPTIONAL DELIMITERS FOR <PERCENTSIGN> MODE I/O.
PRINT @37,0: EOLCHR, ETXCHR, NULCHR

CPYFLG(0008)C DISPLAY COPY SUSPENDED FLAG IN CRTSTT.

CRASH (041A)V HOLDING AREA FOR SYSTEM ERROR DATA.

CRCOD (0094)C CODE FOR CR TOKEN.

CREATE(0500)C CREATE COMMAND I/O SYSTEM CONTROL CONSTANT.

CRECOD(009F)C CODE FOR CRE TOKEN.

CRLF (C278)E ROUTINE TO SEND A CR TO THE OUTPUT BUFFER.

CRLFLF(C276)E ROUTINE TO SEND TWO CR'S TO THE OUTPUT BUFFER.

CROCOD(0084)C CODE FOR CRO TOKEN.

CROS (8F3A)E DISPLAY GRAPHIC POINTER ROUTINE.

CROSS (0048)C BANK SWITCH CONTROL CONSTANT.

CRSTAT(006A)V INPUT BUFFER STATUS FLAG BYTE.

CRTDRV(C634)M DISPLAY DRIVER MODULE.

CRTRST(CBBF)E ROUTINE TO RESTORE THE DISPLAY AFTER A MAGTAPE OPERATION.

CTKN (0053)V CURRENT TOKEN - A HOLDING AREA FOR THE TOKEN BEING EXECUTED.

CTLCHR(F22E)E ROUTINE TO SEND AN ASCII CHARACTER TO THE DISPLAY. IT ALSO HANDLES THE PAGE FULL CONDITIONS.

CURSER(00E6)V RELATIVE CURSOR POINTER.

CURSOR(00B2)V ASCII CODE FOR THE CURSOR CHARACTER.

D Symbols

D5X (AA4C)E DNX - SUBTRACT N FROM X REGISTER.

D6X (AA4B)E " "

D7X (AA4A)E " "

D8X (AA49)E " "

D9X (AA48)E " "

D10X (AA47)E " "

D11X (AA46)E " "

D12X (AA45)E " "

D13X (AA44)E " "

DATACN(BE56)E DATA CONSTANT - ROUTINE TO STACK CONSTANT DURING LINE EXECUTION.

DATCOD(00A5)C CODE FOR DAT TOKEN.

DATIN (CD9E)E ROUTINE TO READ FROM A DATA STATEMENT.

DB (05E0)V DATA BASE POINTER (PRINT USING).

DDP (05F3)V DIGITS DECIMAL POINT.

DEF (9396)E SUBROUTINE TO IMPLEMENT DEF FNX(A)

DEFCOD(00A7)C CODE FOR DEF TOKEN.

DEFPNT(D183)E DEFAULT PRINT.

DEG (C228)E ROUTINE TO IMPLEMENT SET DEGREE STATEMENT.

DEGCOD(0096)C CODE FOR DEG TOKEN.

DEGCON(E9BA)C DEGREE CONSTANT - CONVERSION FACTOR FOR TRIG RANGE REDUCTION. =8/360. DEGCON+8 CONTAINS 360/8.

DELCOD(0058)C CODE FOR DEL TOKEN.

DELET (E6AB)E ROUTINE THAT DELETES PROGRAM LINES.

DELETE(E61E)E ROUTINE THAT PERFORMS THE DELETE STATEMENT.

DELY1S(FEBC)E ROUTINE TO WAIT SOME TIME IN 40USEC. INCREMENTS.

DELY3S(FEB2)E ANOTHER WAIT ROUTINE.

DET (0011)C BANK SWITCH CONTROL CONSTANT.

DETADD(05CE)V HOLDING AREA FOR VALUE OF DETERMINATE.

DETCOD(00A2)C CODE FOR DET TOKEN.

DEXP (00D4)V TWO BYTE BUFFER FOR EXPONENT DURING FLOATING POINT INPUT CONVERSION.

DIGCNT(05F6)V NUMBER OF OUTPUT DIGITS.

DIGFLG(05A7)V FLAG FOR FP INPUT: "NO DIGITS SEEN YET."

DIM (E34A)E DIMENSION TOKEN HANDLER ROUTINE.

DIMCNT(001A)V DIMENSIONSUBSCRIPT COUNTER.

DIMCOD(00A6)C CODE FOR DIM TOKEN.

DIMLP (002C)V DIMENSION LOOP COUNTCONTROL.

DIMSTR(E4D4)E DIMENSION (ALLOCATE MEMORY) FOR A STRING VARIABLE.

DIMSUB(E34A)M DIMENSIONSUBSCRIPT MODULE.

DIR (0900)C DIRECTORY COMMAND I/O SYSTEM CONTROL CONSTANT.

DIRCOD(007D)C CODE FOR DIR TOKEN.

DISBLE(A9CC)E DISABLE INTERRUPTS - ROUTINE TO SET SYSTEM NOT SAFE STATUS.

DISCNT(00B1)V CURSOR GENERATOR COUNT REGISTER.

DISKCL(AB3B)E DISK CALL - ROUTINE TO CALL DISK INTERFACE ROM PACK.

DIVCOD(000F)C CODE FOR DIV TOKEN.

DL (00EB)V DATA LENGTH.

DLTALL(C1A6)E DELETE ALL - ROUTINE TO DO A DELETE ALL.

DLTXS (049A)V DLTXS := XMAX VIEWPORT - XMIN VIEWPORT

DLTYS (04AA)V DLTYS := YMAX VIEWPORT - YMIN VIEWPORT

DN1 (05F4)V NUMBER OF LEFT DIGITS.

DN2 (05F5)V NUMBER OF RIGHT DIGITS.

DOFP (E8DE)E ROUTINE TO INTERPRET FLOATING POINT CODES.

DOLFLG(00F4)V DOLLAR FLAG.

DOTON (00AF)V DOT CONTROL REGISTER FOR CRTDRV.

DP (0008)V DATA POINTER.

DPCONT(C60A)V DECIMAL POINT CONTROL.

DRACOD(0003)C CODE FOR DRA TOKEN.

DRAW (0028)C BANK SWITCH CONTROL CONSTANT.

DRBUSY(C6B4)E ROUTINE TO WAIT FOR DISPLAY TO BE READY.

DREXTA(0058)E DIRECT EXIT A - A JMP INSTRUCTION THAT IS IN RAM FOR DIRTY EXITS FROM ROUTINES THAT MUST PUSH OR PULL PARAMETERS ON THE STACK.

DREXTB(005B)E DIRECT EXIT B - SEE DREXTA.

DSDRAW(F3AE)E ROUTINE TO SEND GDU FPN'S TO DISPLAY VECTORS (DRAW).

DSFLG (0022)V DIMENSIONSUBSCRIPT FLAG.

DSFONT(F42B)E ROUTINE TO SELECT DISPLAY CHARACTER FONT.
PRINT @32,18:FONT

DSFULL(F434)E ROUTINE TO SET UP PAGE FULL FUNCTION.
PRINT @32,26: FULL FUNCTION <0,1,2,3>

DSGRAF(F3B1)E ROUTINE TO SEND GDU FPN'S TO DISPLAY VECTORS.

DSHOME(C6C8)E ROUTINE TO HOME THE DISPLAY CURSOR.

DSMOVE(F3A8)E ROUTINE TO SEND GDU FPN'S TO DISPLAY VECTORS (MOVE).

DSPAGE(C6BE)E ROUTINE TO PAGE THE DISPLAY.

DSPCHR(F2A6)E ROUTINE TO SEND ASCII CHARACTERS TO THE DISPLAY.
NOTE: CONTROL CHARACTERS EXCEPT <CR> WILL
BE CONVERTED TO "LIST FORMAT" BEFORE DISPLAYING.

DSPCP2(F354)E ROUTINE TO INITIATE A DISPLAY COPY.

DSPCPY(F34B)E ROUTINE TO INITIATE A DISPLAY COPY, BUT SET COPY SUSPENDED FLAG IN CRTSTT IF DISPLAY DISABLED.

DSPGIN(F11F)E ROUTINE TO PERFORM DISPLAY INPUT FUNCTIONS.
INPUT AND GIN.

DSPLY (F447)E ROUTINE TO HANDLE THE PAGE AND HOME COMMANDS.

DSPOUT(F2EA)E ROUTINE TO SEND THE OUTPUT BUFFER TO THE DISPLAY IN AN APPROPRIATE MANNER AS DICTATED BY A.SEC.

DSPSTT(0071)V DISPLAY STATUS FLAG BYTE.

DSPVCT(046D)V VECTOR TO OPTIONAL DISPLAY CHARACTER HANDLER. USED BY ROM PACKS AND THE OPT. 1 COMMUNICATIONS.

DT (00ED)V DATA TYPE (1=NUMERIC).

E Symbols

E1 (05E3)V EXPONENT LSD.

E2 (05E4)V EXPONENT DIGIT.

E3 (05E5)V EXPONENT MSD.

EARLYW(00B7)V MAGTAPE DRIVE STATUS FLAG.

EDTBFR(0221)V EDIT BUFFER - A 74 BYTE BUFFER FOR THE LINE EDITOR WORK AREA.

EDTCLR(81FB)E EDIT CLEAR - ROUTINE TO CLEAR THE LINE BUFFER. SETS BUFFER TO SPACES.

EDTCLS(822F)E EDIT CLOSE - ROUTINE TO CLOSE THE EDIT BUFFER. LINE IS NOT LOST.

EDTEND(0060)V EDIT END - A VARIABLE WITH THE END OF EDIT BUFFER ADDRES (EDTBFR+71).

EDTFLG(0478)V USED BY OPTION 1.

EDTMAX(0064)V EDIT MAX - WORKING VARIABLE FOR LINE EDITOR.

EDTPTR(0062)V EDIT POINTER - THE LOCATION OF THE CURSOR IN THE EDIT BUFFER.

EFLAG (05A9)V FLAG SET OF FLOATING POINT INPUT CONVERSION. "SEEN AN E?"

ENABLE(A994)E ENABLE INTERRUPTS - SET THE SYSTEM INTO THE SAFE STATUS.

END (AAB6)E ROUTINE TO IMPLEMENT THE END STATEMENT.

ENDCOD(0060)C CODE FOR END TOKEN.

ENDKEY(0040)C KBFLAG STATUS BIT -- RETURN KEY STRUCK, EDIT BUFFER READY FOR PROCESSING.

ENDTBL(041A)C END OF TABLE - END OF TABLES TO BE CLEARED BY INIT.

EOF (C0AC)E ROUTINE TO IMPLEMENT ON EOF(UNIT) THEN STMT AND OFF EOF(UNIT) STATEMENTS.

EOFCOD(009E)C CODE FOR EOF TOKEN.

EOFTBL(03C9)V EOF TABLE - TABLE TO STORE ACTIVE ON EOF LINE ADDRESSES.

EOICOD(00AD)C CODE FOR EOI TOKEN.

EOIOFF(F576)E ROUTINE TO TURN OF EOI GPIB BIT.

EOION (F56D)E ROUTINE TO TURN ON EOI GPIB BIT.

EOIRDY(F5AB)E ROUTINE TO CLEAR EOI INTERRUPT.

EOL (C12C)E END OF LINE - ROUTINE TO IMPLEMENT LINE TERMINATION FUNCTION.

EOLCHR(0078)V I/O SYSTEM END OF LINE CHARACTER. NORMALLY <CR>.

EOLTG (0018)C END OF LINE TAG - STACK TAG FOR NEW LINE MARK.

EOSTG (0019)C END OF STACK - STACK TAG FOR MARKING THE ABSOLUTE END OF STACK.

EQ (E9EE)E ENTRY FOR FUZZY EQ COMPARE.

EQU (0030)V FLAG SET DURING FP INPUT CONVERSION.

EQUCOD(0011)C CODE FOR EQU TOKEN.

ERAPP1(000E)C APPEND ERROR 1 - BAD ARGUMENTS ON STACK.

ERAPP2(000F)C APPEND ERROR 2 - ATTEMPTING TO APPEND INTO A NONEXISTANT LINE NUMBER.

ERARC (0018)C ERROR CODE FOR INVERSE TRIG FUNCTIONS.

ERASGN(0015)C ASSIGNMENT ARGUMENT INVALID.

ERATSN(0042)C INVALID @<ATSIGN> OR <PERCENTSIGN> SPECIFICATIONS.

ERBCMU(004F)C BAD COMMA USAGE.

ERBDPU(0050)C BAD DECIMAL POINT USAGE.

ERBMDU(004D)C BAD MODIFIER USAGE.

ERBPNU(004C)C BAD PARENTHESIS USAGE.

ERBRK (0028)C ABORT ERROR CODE.

ERCARC(001A)C NOT USED (WE HOPE).

ERCHAR(005E)C BAD CHARACTER ERROR, FROM LEX.

ERCLAI(000C)C ROM-PACK CALL ARGUMENT ERROR.

ERCNSF(0020)C CALL NAME NOT FOUND.

ERCTRA(0435)V CONTAINS TOKEN NUMBER AT WHICH ERROR OCCURED ON ENTRY TO UNLEX.

ERCTRB(0436)V CONTAINS CHARACTER NUMBER AT WHICH ERROR OCCURED ON EXIT FROM UNLEX.

ERDAGN(0056)C USING ERROR -- DATA GONE AFTER PAGE FULL.

ERDATM(0051)C DATA TYPE MISMATCH.

ERDEOR(0053)C EXPONENT OUT OF RANGE FOR 'D' OPERATOR.

ERDET (001F)C MATRIX INVERSION HAD A 0 DETERMINATE.

ERDIM (0009)C DIMENSION ERROR.

ERDOMN(001B)C DOMAIN ERROR.

EREOFN(0014)C EOF OF UNIT N

EREOM (0036)C ERROR -- END OF MEDIUM ON MAGTAPE.

EREXEC(0044)C AN ERROR WAS SEEN BY THE EXECUTE ROUTINE, EXEC.

EREXP (0004)C ERROR CODE FOR EXPONENTIAL - ARGUMENT TOO BIG, RESULT OVERFLOWED.

ERFBFR(0031)C INTERNAL ERROR -- FULL OUTPUT BUFFER WITH NOOUT SET.

ERFILE(003E)C FILE SYSTEM ERROR MESSAGE, INVALID I/O OP.

ERFIXN(005F)C FIX1A ATTEMPTED OF NEGATIVE NUMBER.

ERFLDO(0057)C FIELD OVERFLOW.

ERFNFD(0034)C MAGTAPE ERROR -- FILE NOT FOUND.

ERFORA(0013)C FOR ARGUMENT ERROR.

ERFPDV(0002)C ATTEMPT TO DIVIDE A FLOATING POINT NUMBER BY ZERO.

ERFPOV(0001)C FLOATING POINT RESULT OVERFLOWED.

ERFUZZ(0010)C ERROR IN SPECIFYING FUZZ PARAMETERS.

ERFXOV(0060)C FIX1A ATTEMPTED; RESULT OVERFLOWED.

ERIFC (0049)C ILLEGAL FORMAT CHARACTER.

ERINFS(0048)C OUT OF FORMAT STRING.

ERINTR(0058)C INVALID DATA TYPE TRANSFERED TO AN INTERNAL DEVICE.
PRINT @32,20: "50,65"

```
ERINU (004A)C    ILLEGAL NUMBER USAGE.
ERIOE (0045)C    GPIB BUS ERROR. NOBODY LISTENING (NRFD AND NDAC HIGH).
ERISGN(0054)C    IMAGE STRING GONE AFTER PAGE FULL
ERISTS(0055)C    IMAGE STRING TOO SHORT AFTER PAGE FULL.
ERLNNF(0033)C    REFERENCED LINE NOT FOUND.
ERLNNO(0007)C    LINE NUMBER OUT OF RANGE.
ERLOG (0017)C    ERROR CODE FOR LOG - ARGUMENT ZERO OR NEGATIVE.
ERLOLD(0032)C    LINE TO LONG TO INPUT UNDER OLD OR APP.
ERLRSU(004E)C    BAD LRS USAGE.
ERMARK(0012)C    NOT USED (WE HOPE).
ERMFMT(003A)C    MAGTAPE FORMAT ERROR DETECTED DURING FIND.
ERMILA(0037)C    ILLEGAL MAGTAPE ACCESS.
ERMOPN(003D)C    MAGTAPE FILE OPEN WHEN MARK CALLED.
ERMTRD(0035)C    MAGTAPE READ ERROR.
ERMUL (001E)C    MATRIX MULT. ARGUMENTS INVALID.
ERNCN (0021)C    NOT USED (WE HOPE).
ERNCRT(0039)C    NO MAGTAPE CARTRIDGE.
ERNDT (0022)C    DATA STATEMENT REFERENCE ERROR.
ERNFST(0047)C    NO FORMAT STRING.
ERNIMX(0023)C    INVALID STATEMENT IN IMMEDIATE MODE.
ERNIOD(0043)C    NO I/O DEVICE HANDLER FOR SPECIFIED A.PRIM.
ERNOFN(000B)C    FNN NOT DEFINED.
ERNOLD(003B)C    NOTHING FOUND TO OLD OR APPEND.
ERNON1(0029)C    SIZE ON UNIT.
ERNON2(002A)C    FULL ON UNIT.
ERNON3(002B)C    SRQ ON UNIT.
```

```
ERNON4(002C)C    EOI ON UNIT.

ERNON5(002D)C    EXTERNAL INTERRUPT  1.

ERNON6(002E)C    EXTERNAL INTERRUPT  2.

ERNON7(002F)C    EXTERNAL INTERRUPT  3.

ERNON8(0030)C    EOF N ON UNIT.

ERNSEP(0041)C    NOT USED.

ERNXFN(0025)C    EXTERNAL FUNCTION ROM REQUIRED.

ERNXTA(0013)C    NEXT STATEMENT INVALID.

EROFA (0013)C    GOTO N OF ... OR GOSUB N OF ... WHERE N IS INVALID.

EROVRD(003C)C    ILLEGAL MAGTAPE RECORD LENGTH ENCOUNTERED.

EROUOT(0046)C    INCOMPLETE LITERAL.

ERRCD (0045)V    ERROR CODE - HOLDING AREA FOR ERROR CODE.  IF ZERO NO
                 ERRORS HAVE OCCURED IN THIS LINE.

ERRCDB(004C)V    ERROR CODE B - BACK UP HOLDING AREA FOR ERRCD WHILE
                 PROCESSING SIZE ERRORS.

ERREAD(003F)C    BINARY HEADER ERROR.

ERREN (0011)C    RENUMBER ERROR.

ERREP (001C)C    REP FUNCTION ARGUMENT IS BAD.

ERROM1(0059)C    ANY ERROR IN ROM PACK.

ERROM2(005A)C    STOP EVAL ONLY -- ROM PACK USAGE.

ERSEC (0026)C    PROGRAM SECRET.

ERSFE (0019)C    CHR N --- N OUT OF RANGE.

ERSHAP(0008)C    MATRICIES ARE NOT CONFORMABLE.

ERSQR (0006)C    ERROR CODE FOR SQUARE ROOT.

ERSTGL(0040)C    NOT USED.

ERSTOP(005B)C    STOP ERROR CODE.

ERSUBC(000A)C    SUBSCRIPT ERROR.
```

ERSYNX(005D)C PARSER DOWN ARROW.

ERTABO(0052)C TABBING ERROR.

ERTERM(005C)C PROGRAM ABORTED WITH BREAK KEY.

ERTMDS(004B)C TOO MANY DATA SPECIFIERS.

ERTRNG(0005)C ERROR CODE: TRIGONOMETRIC ARGUMENT TOO LARGE.

ERUNDF(0024)C ARGUMENT UNDEFINED.

ERUP (0003)C ERROR CODE FOR 0**0.

ERUPN (0016)C ERROR CODE FOR (NEG)**B.

ERVAL (001D)C VAL A$ -- NO VALUE FOUND IN A$.

ERWBYT(000D)C ILLEGAL VALUE SPECIFIED FOR WBYTE.

ERWPT (0038)C MAGTAPE WRITE LOCKED.

ERWSFL(0027)C WORKSPACE FULL ERROR (MEMORY FULL).

ES (05E2)V ASCII EXPONENT SIGN.

ESTG (0002)C EVALUATOR STATUS TAG - STACK TAG FOR EVAL STATUS.

ETOX (EC4C)E COMPUTES FLOATING POINT EXPONENTIAL FUNCTION E**X.

ETXCHR(0079)V I/O SYSTEM END OF TEXT CHARACTER (EOF)

EVLEN (9368)M EVAL ENTRY POINTS - MAIN ENTRY POINTS AND RECURSION POINTS FOR EVAL.

EVLSUB(CE70)M EVAL SUBROUTINES - EVAL SERVICE SUBROUTINES.

EXEC (F45C)E ASCII TO HEX LOAD ROUTINE: LOADS STARTING AT SCRTCH.

EXH (00D4)V DECIMAL EQUIVALENT EXPONENT, HIGH ORDER.

EXL (00D5)V DECIMAL EQUIVALENT EXPONENT, LOW ORDER.

EXP (05F9)V CONDENSED EXPONENT.

EXPCOD(0046)C CODE FOR EXP TOKEN.

EXS (05FA)V EXPONENT BINARY SIGN.

F1 (05F7)V "AS REQUIRED" LEFT OF DECIMAL POINT.

F2 (05F8)V "AS REQUIRED" RIGHT OF DECIMAL POINT.

FA (0090)C TAG FOR DOFP: ADD TWO WORDS ON STACK.

FART (001C)C TAG FOR DOFP: NO-OP USED WHEN ONLY FA,FD,FM, OR FS IS TO BE PERFORMED.

FD (00EC)C TAG FOR DOFP: CIVIDE TWO WORDS ON STACK.

FDUP (0016)C TAG FOR DOFP: DUPLICATE TOP FLOATING POINT STACK ENTRY.

FFLG (00F1)V "AS REQUIRED" FLAG.

FFOT (0004)C MAGTAPE DRIVER FLAG --- NOT USED.

FHEX (0000)C TAG FOR DOFP: FLOATING POINT PUSH EXTENDED.

FHIM (001D)C TAG FOR DOFP: FLOATING POINT PUSH IMMEDIATE.

FHIN (0008)C TAG FOR DOFP: FLOATING POINT PUSH INDIRECT.

FIL (E870)E ROUTINE TO HANDLE .

FILCOD (004F)C CODE FOR FIL TOKEN.

FILEIN (E858)E ROUTINE TO ESTABLISH LINKAGE TO FILE SYSTEM FOR INPUT.

FILEOT (E868)E ROUTINE TO ESTABLISH LINKAGE TO FILE SYSTEM FOR OUTPUT.

FILERQ (E876)E ROUTINE TO ESTABLISH LINKAGE WITH FILE SYSTEM DIRECTLY.

FILES (E86C)E ROUTINE TO ESTABLISH LINKAGE TO FILE SYSTEM FOR THE COMMAND COMMAND.

FILFND (00BD)V MAGTAPE FILE TO BE FOUND REGISTER.

FILLOC (00BB)V PRESENT MAGTAPE FILE LOCATION REGISTER.

FINCOD (0076)C CODE FOR FIN TOKEN.

FIND (1B21)C FIND COMMAND I/O SYSTEM CONTROL CONSTANT.

FIX1 (B1A2)E ROUTINE TO GET INTEGER PART OF FLOATING POINT NUMBER.

FIXNUM (B08F)E I/O ROUTINE TO FIX I/O LIST ENTRIES FOR INTERNAL USE.

FLAG (0032)V USED AS A COUNTER FOR LITERAL LENGTH IN ROUTINE UNCOMP.

FLAGS (00F7)V STATUS FLAG BYTE FOR PRINTF.

FLCTRL(9F16)M FLOW OF CONTROL - EVAL ROUTINES FOR FLOW OF CONTROL STATEMENTS. GOTO, GOSUB, OF, RUN, STOP, RETURN, IF, FOR AND NEXT.

FLDATA(BE56)M FLOW OF DATA - EVAL ROUTINES FOR STACKING CONSTANTS AND VARIABLES AND FOR MOVING DATA FOR USER DEFINED FUNCTIONS AND ASSIGN.

FLEX (000D)C TAG FOR DOFP: FLOATING POINT POP EXTENDED.

FLIN (0013)C TAG FOR DOFP: FLOATING POINT POP INDIRECT.

FLOAT1(B1FB)E SAME AS FLOAT2.

FLOAT2(B1FB)E ROUTINE TO CONVERT INTEGER TO FLOATING POINT.

FLUNIT(000B)C BANK CONTROL CONSTANTS FOR ON PAGE AND OFF PAGE STATEMENTS.

FM (0000)C TAG FOR DOFP: MULTIPLY TWO WORDS ON STACK.

FMTCLN(D208)E FORMAT STRING CLEAN-UP ROUTINE.

FMTFLG(00F6)V FORMAT TYPE IN PROCESS (0=FMTOUT).

FMTINI(D107)E FORMAT INITIALIZE ROUTINE.

FMTPNT(D178)E FORMAT DATA ITEM OUTPUT ROUTINE.

FNACOD(0024)C CODE FOR FNA TOKEN.

FNASGN(BEE1)E FUNCTION ASSIGN - ROUTINE TO DO ASSIGNMENT IN FNX LINES.

FNBCOD(0025)C CODE FOR FNB TOKEN.

FNCCOD(0026)C CODE FOR FNC TOKEN.

FNDCOD(0027)C CODE FOR FND TOKEN.

FNECOD(0028)C CODE FOR FNE TOKEN.

FNEVL (9368)E FUNCTION EVALUATION - ROUTINE TO IMPLEMENT FNX(PARM) CALLS.

FNFCOD(0029)C CODE FOR FNF TOKEN.

FNGCOD(002A)C CODE FOR FNG TOKEN.

FNHCOD(0026)C CODE FOR FNH TOKEN.

FNICOD(002C)C CODE FOR FNI TOKEN.

FNJCOD(002D)C CODE FOF FNJ TOKEN.

FNKCOD(002E)C CODE FOR FNK TOKEN.

FNLCOD(002F)C CODE FOR FNL TOKEN.

FNMCOD(0030)C CODE FOR FNM TOKEN.

FNNCOD(0031)C CODE FOR FNN TOKEN.

FNOCOD(0032)C CODE FOR FNO TOKEN.

FNPARM(BEC1)E FUNCTION PARAMETER - ROUTINE TO FETCH PARAMETER FOR FNX USAGE.

FNPCOD(0033)C CODE FOR FNP TOKEN.

FNQCOD(0034)C CODE FOR FNQ TOKEN.

FNRCOD(0035)C CODE FOR FNR TOKEN.

FNSCOD(0036)C CODE FOR FNS TOKEN.

FNTBL (03DD)V FUNCTION TABLE - TABLE TO STORE ADDRESSES OF FNX LINES.

FNTCOD(0037)C CODE FOR FNT TOKEN.

FNUCOD(0038)C CODE FOR FNU TOKEN.

FNVCOD(0039)C CODE FOR FNV TOKEN.

FNWCOD(003A)C CODE FOR FNW TOKEN.

FNXCOD(003B)C CODE FOR FNX TOKEN.

FNYCOD(003C)C CODE FOR FNY TOKEN.

FNZCOD(003D)C CODE FOR FNZ TOKEN.

FONT (00B3)V PRESENT DISPLAY FONT.

FOR (A026)E ROUTINE TO IMPLEMENT THE FOR STATEMENT.

FORCOD(0050)C CODE FOR FOR TOKEN.

FORTG (0004)C FOR TAG - STACK TAG FOR FOR/NEXT STATMENT WORK AREA.

FPA (05AA)V 8-BYTE BUFFER TO HOLD FP NUMBERS.

FPADD (B245)E ROUTINE TO DO FP ADD.

FPAITT(B014)E ROUTINE TO CONVERT FPN TO ASCII. INTERNAL BUFFERS.

FPC (053A)V 8-BYTE BUFFER FOR FP NUMBERS.

FPCMP (EE3E)E ROUTINE TO COMPARE TWO FP NUMBERS ON STACK.

FPDIV (B303)E ROUTINE TO DO FLOATING POINT DIVIDE.

FPMUL (B4AE)E ROUTINE TO DO FLOATING POINT MULTIPLY.

FPNASC(B853)E ROUTINE TO REDUCE A FP NUMBER TO A FRACTION FROM 0.1 TO 1.0 FOR CONVERSION TO ASCII LATER.

FPNEG (EBAA)E NEGATES FP NUMBER ON TOP OF STACK.

FPONE (E950)C FLOATING POINT 1.0

FPSUB (B23E)C ROUTINE TO DO FP SUBTRACT.

FPTWO (E92E)C FLOATING POINT 2.0

FPZERO(E926)C FLOATING POINT 0.0

FRES (0019)C TAG FOR DOFP: COPY FP NUMBER FROM FPA TO TOP OF STACK.

FRET (000B)C TAG FOR DOFP: END INTERPRETATION AND RETURN TO CALLER.

FS (00A0)C TAG FOR DOFP: SUBTRACT TWO WORDS ON TOP OF STACK.

FSAV (0018)C TAG FOR DOFP: POP FP NUMBER FROM TOP OF STACK TO FPA.

FSWP (0012)C TAG FOR DOFP: SWAP TOP TWO FP ENTRIES ON STACK.

FUDGH (0031)C FUDGE - HIGH BYTE OF STACK SPACE REQUIRED TO EXECUTE A LINE CONSTANT.

FUDGL (00F4)C LOW BYTE OF FUDGE CONSTANT.

FULCOD(00AB)C CODE FOR FUL TOKEN.

FULL (00B0)V INDICATES WHEN DISPLAY BECOMES FULL.

FULSCN(F2C2)E ROUTINE TO SCAN "BLINKING F" WHILE WAITING FOR PAGE KEY.

FULSTT(058E)V INDICATES WHAT ACTION IS TO BE TAKEN WHEN DISPLAY BECOMES FULL.

FUZA (05C2)V CONTAINS FUZZ FACTORE FOR NON-ZERO COMPARES.

FUZB (05C3)V CONTAINS EXPONENT FOR FUZZY COMPARE TO ZERO.

FUZCOD(0080)C CODE FOR FUZ TOKEN.

FUZR (05CB)V VARIABLE OFR FUZZ: SET IN NORM: EQUALS NUMBER FO SHIFTS REQUIRED TO NORMALIZE FRACTION.

FUZZIE(B80D)E ROUTINE TO DO FUZZY COMPARE.

G Symbols

GENCUR(F36F)E ROUTINE TO GENERATE A BLINKING CURSOR.

GETCHR(F085)E ROUTINE TO GET NEXT CHARACTER FOR FP INPUT CONVERSION.

GETKEY(F7DC)E ROUTINE TO HANDLE THE KEYBOARD HARDWARE.

GETLAR(9106)E ROUTINE TO FIND THE ADDRESS AND LINE NUMBER OF THE LINE WITH THE SMALLEST LINE NUMBER LARGER THAN OR EQUAL TO THE ARGUMENT PASSED ON THE STACK.

GETLIN(9088)M MODULE CONTAINING GETSMA AND GETLAR.

GETLN (0087)E GET LINE - ROUTINE TO CONVERT FLOATING POINT NUMBER ON THE STACK TO THE ADDRESS OF THE GIVEN LINE.

GETLNA(0082)E GET LINE A - ROUTINE TO CONVERT INTEGER IN R0 TO THE ADDRESS OF THE GIVEN LINE.

GETSMA(90F8)E ROUTINE TO FIND THE ADDRESS AND LINE NUMBER OF THE LINE WITH THE LARGEST LINE NUMBER SMALLER THAN OR EQUAL TO THE ARGUMENT PASSED ON THE STACK.

GIN (0050)C BANK SWITCH CONTROL CONSTANT.

GINCOD(0063)C CODE FOR GIN TOKEN.

GINSET(92ED)E ROUTINE TO SET UP POINTERS FOR GIN INPUT.

GLBFLG(0055)V GLOBAL FLAGS - EVAL GLOBAL CONTROL FLAGS.

GOCOD (0052)C CODE FOR GO TOKEN.

GOSCOD(0053)C CODE FOR GOS TOKEN.

GOSTG (0003)C GOSUB TAG - STACK TAG FOR GOSUB/RETURN ENTRY.

GOSUB (9F60)E ROUTINE TO IMPLEMENT THE GOSUB STATEMENT.

GOTO (9F52)E ROUTINE TO IMPLEMENT TO GOTO STATEMENT.

GRACOD(0095)C CODE FOR GRA TOKEN.

GRAD (C22D)E ROUTINE TO IMPLEMENT SET GRAD COMMAND.

GRAF (8FFE)M GRAPHICS CONTROL MODULE.

GRAPH (9238)E ROUTINE TO HANDLE DRAW, MOVE, RDRAW, RMOVE, GIN.

GRDCON(E9CA)C GRAD CONSTANT - CONSTANT FOR TRIG RANGE REDUCTION. =8/400
GRDCON+8 = FP CONSTANT 400/8

GRFDR.(9469)E ROUTINE TO PERFORM CLIPPING ON VECTORS.

I Symbols

IDLE (C5B2)E ROUTINE THAT IS MASTER IDLE LOOP. ALL CONTROL IS PASSED FROM HERE ONCE THE SYSTEM HAS BEEN POWERED UP.

IDLVCT(0470)V VECTOR TO OPTIONAL ROUTINE TO GAIN CONTROL FROM MASTER IDLE LOOP.

IEC (AF82)E ROUTINE TO HANDLE @<ATSIGN>.

IECCOD(004E)C CODE FOR IEC TOKEN.

IECDRV(0001)M GPIB (IEC) CONTROL MODULE.

IECEOI(F591)E ROUTINE TO SERVICE EOI HARDWARE INTERRUPTS.

IECIFC(F55A)E ROUTINE TO GENERATE IFC FUNCTION ON GPIB.

IECIN (F60F)E ROUTINE TO RECEIVE A BUFFER FROM THE GPIB.

IECOFF(F408)E ROUTINE TO INITIALIZE GPIB HARDWARE TO IDLE STATE.

IECOUT(F5C0)E ROUTINE TO SEND A BUFFER ALONG THE GPIB.

IECRD (F4E3)E ROUTINE TO READ ONE CHARACTER FROM THE GPIB.

IECRLF(0463)V INDICATES CR VS. CR/LF FLAG STATUS.

IECSND(F50E)E ROUTINE TO SEND ONE CHARACTER ALONG THE GPIB.

IF (A013)E ROUTINE TO IMPLEMENT THE IF STATEMENT.

IFCOD (0054)C CODE FOR IF TOKEN.

IMACOD(00A3)C CODE FOR IMA TOKEN.

IMMTBL(A6A9)C IMMEDIATE MODE TABLE IN PARSE.

IMXTG (0007)C IMMEDIATE EXECUTE TAG - STACK TAG FOR ADDRESS OF A LINE.

INAGTE(B0C2)E SPECIALIZED ROUTINE TO CONVERT INTEGER TO ASCII.

INAITT(B004)E GENERAL ROUTINE TO CONVERT INTEGER TO ASCII. INTERNAL BUFFER.

INCXN (A2CC)E INCREMENTS INDEX REGISTER TO THE NEXT LEX TOKEN.

INFO (90A6)E RETURNS THE INFORMATION IN OPTABLE FOR THE SPECIFIED 4051 BASIC TOKEN.

INICOD(0062)C CODE FOR INI TOKEN.

INIT (C143)E ROUTINE TO IMPLEMENT INIT COMMAND.

INITMT(E00C)E ROUTINE TO INITIALIZE MAGTAPE HARDWARE AND DISABLE DISPL

INITX (C146)E SPECIAL ENTRY POINT TO INIT FOR POWER UP AND DELETE ALL ROUTINES TO USE.

INPCOD(0069)C CODE FOR INP TOKEN.

INPCTL(EFF8)M INPUT CONTROL MODULE.

INPNT (0098)V INPNT := A.PTR

INPSTG(EFF8)E ROUTINE TO INPUT A STRING.

INPUT (0D9F)C INPUT COMMAND I/O SYSTEM CONTROL CONSTANT.

INPVAL(F033)E ROUTINE TO INPUT A VALUE.

INT (EF3C)E ROUTINE TO GET INTEGER PART OF FPN AND RETURN AS FPN.

INT1 (0016)V INTEGER WORKING STORAGE.

INT2 (0018)V INTEGER WORKING STORAGE.

INTACP(F480)E ROUTINE TO SET GPIB HARDWARE TO LISTEN STATE.

INTCN (BE84)E INTEGER CONSTANT - ROUTINE TO FLOAT AND STACK INTEGER.

INTCOD(0043)C CODE FOR INT TOKEN.

INTMLA(E5FB)E INTEGER MULTIPLY ROUTINE WITH ACCUMULATE.

INTMLC(E5F9)E INTEGER MULTIPLY ROUTINE, NO ACCUMULATE.

INTQQQ(005E)V INTERRUPT ROUTINE TEMPORARY STORAGE.

INTSRC(F539)E ROUTINE TO INITIALIZE GPIB HARDWARE TO SOURCE STATE.

INTSRV(EFB8)E HARDWARE INTERRUPT SERVICE ROUTINE. DISPATCHES TO INDIVIDUAL HANDLERS USING THE PIATBL.

INV (0019)C BANK SWITCH CONTROL CONSTANT.

INVCOD(001B)C CODE FOR INV TOKEN.

IOBFR1(0268)C BEGINNING OF GENERAL I/O BUFFER.

IOCLNR(AB67)E ROUTINE TO CLEAN UP STACK AFTER I/O PROCESSING.

IOFLGS(008A)V I/O PROCESSOR STATUS FLAG BYTE.

IOFUNC(008E)V PRESENT I/O STATEMENT CODE FOR I/O PROCESSOR.

IOKONS(E61E)M MODULE CONTAINING THE DEFINITION OF ALL I/O SYSTEM CONTROL CONSTANTS.

IOPROC(AB5C)E ROUTINE TO PROCESS AN I/O LIST. PRINT, INPUT, READ, WRITE, RBYTE, WBYTE, FIND, MARK, KILL.

IOSCAN(AB86)E ACTUAL I/O LIST SCANNER.

IOSYSF(0467)V I/O SYSTEM ADDRESSED FLAG. USED BY ROM PACKS.

ISB (05DE)V IMAGE STRING BASE (PRINT USING)

ISL (00E9)V IMAGE STRING LENGTH.

ISP (00D6)V IMAGE STRING POINTER.

ITGCOD(00B2)C CODE FOR ITG TOKEN.

ITM1TG(000E)C ITEM ONE TAG - STACK TAG FOR AN INTEGER.

ITM2TG(000F)C ITEM TWO TAG - STACK TAG FOR AN INTEGER.

ITSINT(05A8)V FLAG FOR FP INPUT: "IT'S AN INTEGER."

J Symbols

```
JMPAX  (0450)E    SELF MODIFYING ROUTINE TO JMP TO (ACC-A)+(X).
                  JMPAX: STA A JMPAX+4  :MODIFY DISP FIELD
                   JMP 0,X  :JUMP TO ACC-A+X

JMPX   (043E)E    SELF MODIFYING ROUTINE TO ((ACC-0)+(X)).
                  JMPX: STA A JMPX+4  ;MODIFY DISP FIELD
                   LDX 0,X  ;FETCH JUMP ADDRESS
                   JMP 0,X  ;EXIT

JUNK   (8F1B)C    LABEL ON THE BEGINNING OF OPTABLE.
```

K Symbols

```
K0      (0000)V     R0.
K1      (0001)V     R0+1.
K2      (0002)V     R1.
K3      (0003)V     R1+1.
K4      (0004)V     R2.
K5      (0005)V     R2+1.
K6      (0006)V     R3.
KBFLAG  (006B)V     KEYBOARD PROCESSOR STATUS FLAG BYTE.
KBIN    (0076)V     VALUE OF LAST KEY TRANSFERED BY THE KEYBOARD.
KBINT   (C4DE)E     ROUTINE TO PROCESS KEYBOARD HARDWARE INTERUPT.
KBMRK   (059A)V     POINTER INTO KEYBOARD PSEUDO STACK.
KBPROC  (C590)E     ROUTINE TO PROCESS KEYS AT THE PROGRAM LEVEL.
KBQIN   (C69A)E     ROUTINE TO STUFF A KEY INTO THE TYPE-AHEAD QUE.
KBQOUT  (C64A)E     ROUTINE TO FETCH A KEY FROM THE TYPE-AHEAD QUE.
KBSTK   (059C)V     KEYBOARD DRIVER PSEUDO STACK.
KERNEL  (05D6)V     FP NUMBER WHICH CONTAINS KERNEL OF RANDOM NUMBER GENERATOR.
KEY     (C240)E     ROUTINE TO IMPLEMENT SET KEY.
KEYCNT  (0599)V     DELAY COUNT FOR "OLDEST" KEY.
KEYCOD  (009A)C     CODE FOR KEY TOKEN.
KEYDRV  (F7DC)E     ROUTINE TO DECODE KEYBOARD HARDWARE INTO ASCII KEYS.
KEYIN   (F163)E     ROUTINE TO DO INPUT FROM KEYBOARD.
KEYQUE  (0100)C     BEGINNING OF KEYBOARD TYPE-AHEAD QUE.
KEYSTK  (0411)V     KEY STACK - USED TO STORE UNPROCESSED USER DEFINABLE
                    KEYS.  INPUTS FROM KEYBOARD DRIVER,  OUTPUTS IN EVAL.
KEYTIM  (0598)V     PRESENT KEY DELAY TIME REGISTER.
KILCOD  (0071)C     CODE FOR KIL TOKEN.
KILL    (D9DA)E     ROUTINE TO HANDLE KILL COMMAND.
KILTYP  (0721)C     KILL COMMAND I/O SYSTEM CONTROL CONSTANT.
KYRATE  (05A5)V     MAXIMUM KEYBOARD RATE.
```

L Symbols

```
L1      (00DA)V    REPEAT LOOP 1.

L2      (00DD)V    REPEAT LOOP 2.

L3      (00E0)V    REPEAT LOOP 3.

L4      (00E3)V    REPEAT LOOP 4.

LBKCOD  (0093)C    CODE FOR LBK TOKEN.

LBRKTG  (0010)C    LEFT BRACKET TAG - STACK TAG FOR LEFT BRACKET IN
                   OBJECT STRING.

LCLFLG  (0054)V    LOCAL FLAGS - EVAL FLAGS FOR ONE LINE.

LCOM    (05FF)V    COMMA COUNT.

LDAX    (044B)E    SELF MODIFYING ROUTINE TO LOAD ACC-A FROM (ACC-A)+(X).
                   LDAX: STA A LDAX+4  ;MODIFY DISP FIELD
                    LDA A 0,X  ;GET DATA BYTE
                    RTS   ;EXIT

LDBX    (0451)E    SELF MODIFYING ROUTINE TO LOAD ACC-B FROM (ACC-A)+(X).
                   LDBX: STA A LDBX+4  ;MODIFY DISP FIELD
                    LDA B 0,X  ;GET DATA
                    RTS   ;EXIT

LDIG    (05FC)V    LEFT DIGIT COUNT.

LDIGA   (05FD)V    LEFT DIGIT BEGINNING ADDRESS.

LDXX    (0445)E    SELF MODIFYING ROUTINE TO LOAD X FROM (ACC-A)+(X).
                   LDXX: STA A LDXX+4  ;MODIFY DISP FIELD
                    LDX 0,X  ;GET DATA
                    RTS

LEN     (0068)C    BANK SWITCH CONTROL CONSTANT.

LENCOD  (0020)C    CODE FOR LEN TOKEN.

LENGTH  (009A)V    DIMENSIONED STRING LENGTH FOR I/O PROCESSING OF STRINGS.

LETCOD  (0065)C    CODE FOR LET TOKEN.

LEX     (8826)E    LEXICAL ANALYZER. CONVERTS INPUT LINES FROM ASCII
                   TO INTERNAL FORMAT.

LEXCNT  (0036)V    VARIABLE RETURNED TO TRANSL BY LEX EQUAL TO THE
                   NUMBER OF BYTES OUTPUT INTO SCRTCH BY LEX.

LGN     (ED27)E    ROUTINE TO DO NATURAL LOGARITHM.
```

LGNCOD(0044)C TOKEN CODE FOR LGN.

LINCN (BE84)E LINE CONSTANT - ROUTINE TO FLOAT A LINE NUMBER AND STACK IT.

LINCOD(00B3)C CODE FOR LIN TOKEN.

LINELN(058F)V DISPLAY LINE LENGTH. MODIFIED BY OPT. 1 COMMUNICATIONS.

LISCOD(005A)C CODE FOR LIS TOKEN.

LIST (1320)C LIST COMMAND I/O SYSTEM CONTROL CONSTANT.

LISTTG(0005)C LIST TAG - STACK TAG FOR LIST COMMAND ENTRY.

LITCN (BE69)E LITERAL CONSTANT - ROUTINE TO PUSH POINTER TO LITERAL.

LITCOD(0095)C CODE FOR LIT TOKEN.

LITFLG(00EE)V LITERAL FLAG.

LITREL(9598)E LITERAL RELATION - ROUTINE TO IMPLEMENT STRING COMPARE.

LNEVL (963C)E LINE EVALUATOR - ROUTINE TO EVALUATE ONE LINE.

LNNOTG(000D)C LINE NUMBER TAG - STACK TAG FOR SAVED LINE NUMBER.

LOCTG (D0E9)E LOCATE TAG - ROUTINE TO SEARCH FOR A GIVEN TAG.

LOCTGR(D104)E LOCATE TAG IN A RANGE - ROUTINE TO SEARCH THE STACK FOR A TAG IN A RANGE OF VALUES.

LOG (ED27)E ROUTINE TO COMPUTE LOG TO BASE TEN.

LOG2 (E93E)C FP NATURAL LOG OF 2.0

LOGCOD(0045)C CODE FOR LOG TOKEN.

LPNCOD(0092)C CODE FOR LPN TOKEN.

LRSFLG(00EF)V <CR> SUPRESS FLAG FOR PRINTF.

LSP (0045)V LOW STACK POINTER - TOP OF STACK POINTER FOR LOW RAM STACK. THIS IS WORKING SPACE ALLOCATION AREA.

LSPS (05FB)V LEADING SPACES COUNT FOR PRINTF.

LSTCOD(00B6)C CODE FOR LST TOKEN.

LSTKEY(006C)V LAST VALID KEYCODE --- I DON'T THINK IT'S USED.

LT (E9FA)E ENTRY FOR FUZZY LT COMPARE.

```
LTCOD  (0013)C    CODE FOR LT TOKEN.
LTE    (E9F6)E    ENTRY FOR FUZZY LT.EQ COMPARE.
LTECOD(0014)C     CODE FOR LTE TOKEN.

LUNNO  (007D)V    FOR USE BY FILE SYSTEM. ACTIVE LUN.
```

M Symbols

MAGEND(006F)V MAGTAPE DRIVER FLAG TO INDICATE MAGTAPE BUSY.

MARCOD(0079)C CODE FOR MAR TOKEN.

MARK (1C21)C MARK COMMAND I/O SYSTEM CONTROL CONSTANT.

MARKS (FC73)E ROUTINE TO MARK NEW MAGTAPE FILES.

MASK (0039)V UNLEX FLAG, IF NONZERO INDICATES THAT THE LAST CHARACTER OUTPUT WAS A SPACE.

MATCOD(0066)C CODE FOR MAT TOKEN.

MATMAT(94DB)E MATRIX-MATRIX - ROUTINE TO APPLY A SCALER OPERATOR OVER TWO ARRAYS.

MATMOV(BF8B)E MATRIX MOVE - ROUTINE TO ASSIGN ONE MATRIX TO ANOTHER.

MATOPR(94CA)M MATRIX OPERATORS - MODULE WITH MATRIX FUNCTIONS IN IT.

MATSCL(94E6)E MATRIX-SCALER - ROUTINE TO APPLY A SCALER TO AN ARRAY AND SCALER.

MATSIZ(CFFA)E ROUTINE TO CALCULATE CONTROL CONSTANTS FOR A MATRIX.

MAX (EF2C)E FINDS MAX OF TWO FP NUMBERS ON STACK.

MAXANS(B3A1)E ROUTINE TO PUSH LARGEST FP NUMBER ON STACK.

MAXCOD(0009)C CODE FOR MAX TOKEN.

MEMCOD(00A1)C CODE FOR MEM TOKEN.

MEMORY(C1FB)E ROUTINE TO IMPLEMENT THE MEMORY OPERATOR.

MFLG (00F3)V MINUS FLAG.

MIN (EF1B)E FINDS MINIMUM OF TWO FP NUMBERS ON STACK.

MINCOD(0008)C CODE FOR MIN TOKEN.

MKFILE(DB0A)E ROM PACK ENTRY FOR MARKING MAGTAPE FILES.

MMICOD(0022)C CODE FOR MMI TOKEN.

MNSCOD(000D)C CODE FOR MNS TOKEN.

MODMTH(DA02)E ROUTINE TO MODIFY MAGTAPE HEADERS.

MODT88(D9FF)E SPECIAL ROUTINE TO MODIFY MAGTAPE HEADERS.

MOVCOD(0001)C CODE FOR MOV TOKEN.

```
MOVE   (0038)C    BANK SWITCH CONTROL CONSTANT.
MPLCOD(0021)C     CODE FOR MPL TOKEN.
MPY    (0029)C    BANK SWITCH CONTROL CONSTANT.
MPYCOD(0018)C     CODE FOR MPY TOKEN.
MSKCTR(050A)V     SYSTEM MASK COUNTER (ENABLE/DISABLE)
MTAFIN(DBDD)E     ROUTINE TO LOACATE AND OPEN A NEW MAGTAPE FILE.
MTBFR  (011F)C    BEGINNING OF MAGTAPE RECORD BUFFER.
MTBINT(DDF5)E     ROUTINE TO INITIALIZE MAGTAPE BUFFER POINTERS.
MTBSWP(DDFE)E     ROUTINE TO SWAP MAGTAPE BUFFER POINTERS.
MTCHKS(DE21)E     ROUTINE TO COMPUTE CHECKSUM ON A MAGTAPE RECORD.
MTCLOS(DD18)E     ROUTINE TO CLOSE MAGTAPE FILES.
MTDRV  (FB90)M    MAGTAPE HARDWARE CONTROL MODULE.
MTEOF  (0001)C    MAGTAPE DRIVE END-FILE FLAG BIT.
MTERR  (0590)V    MAGTAPE RE-READ COUNTER.
MTFFST(0080)C     MTSTT2 FLAG BIT INDICATES HEAD OVER A GAP.
MTFIND(DBD1)E     ROUTINE TO LOACATE AND OPEN A NEW MAGTAPE
                  FILE GIVEN AN FPN.
MTFLGS(0089)V     MAGTAPE DRIVER HARDWARE FLAGS.
MTFPOS(0040)C     MTSTAT FLAG BIT -- SET IF OPEN AND ACCESSED (READ/WRITE).
MTFSIZ(0080)C     MTSTAT FLAG BIT -- SET IF OPEN FILE IS 128. BYTE
                  RECORD FORMAT, ELSE 256. BYTE ASSUMED.
MTKILL(DA64)E     ROUTINE TO PERFORM KILL N ON THE MAGTAPE.
MTMARK(DAA0)E     ROUTINE TO MARK A NEW FILE.
MTMASK(0468)V     NOT USED.
MTMAX  (0083)V    POINTER TO LAST CHARACTER IN MAGTAPE BUFFER.
MTNULL(DD0C)E     ROUTINE TO NULL A MAGTAPE BUFFER.
MTOPEN(0020)C     MTSTAT FLAG BIT -- SET IF MAGTAPE FILE IS OPEN.
```

MTPADR(DFE8)E ROUTINE TO ADDRESS, SET UP, MAGTAPE SYSTEM.

MTPIN (DF35)E ROUTINE TO GET A LOGICAL BUFFER FROM THE MAGTAPE BUFFER.

MTPINR(DE53)E MAGTAPE INPUT REQUEST VALIDITY INTERPRETER.

MTPINW(DE31)E MAGTAPE OUTPUT FUNCTION INTERPRETER.

MTPOUT(DD98)E ROUTINE TO SEND A BUFFER TO THE MAGTAPE BUFFER.

MTPRWD(C506)E ROUTINE TO DO A CONTROLLED REWIND OF THE MAGTAPE.

MTPTR (0085)V MOVING MAGTAPE BUFFER POINTER.

MTRBFR(DF43)E ROUTINE TO DO A CONTROLLED READ OF A MAGTAPE BUFFER.

MTREAD(DCA8)E ROUTINE TO READ A PHYSICAL MAGTAPE RECORD.

MTSASC(0008)C MTSTAT FLAG BIT -- PRESENT FILE IS ASCII

MTSBIN(0004)C MTSTAT FLAG BIT -- PRESENT FILE IS BINARY

MTSCRT(DFE1)E ROUTINE TO SET MAGTAPE SECRET FILE BIT.

MTSNEW(0010)C MTSTAT FLAG BIT -- PRESENT FILE IS NEW.

MTSPGM(0002)C MTSTAT FLAG BIT -- PRESENT FILE IS PROGRAM.

MTSRD (0010)C MTSTT2 FLAG BIT -- PRESENT MAGTAPE BUFFER VALID FOR READ.

MTSREG(0081)V MAGTAPE FORMAT STATUS REGISTER.

MTSSEC(0001)C MTSTAT FLAG BIT -- PRESENT MAGTAPE FILE IS SECRET.

MTSSET(DFBB)E ROUTINE TO SET THE MAGTAPE FORMAT STATUS REGISTER (MTSREG).
PRINT @33,0: LENGTH, CHECKSUM, HEADER

MTSTAT(0082)V MAGTAPE CONTROLLER STATUS FLAG BYTE.

MTSTT2(0070)V MAGTAPE CONTROLLER STATUS FLAG BYTE.

MTSWRT(0002)C MTSTT2 FLAG BIT -- PRESENT MAGTAPE BUFFER CONTAINS WRITTEN INFORMATION.

MTWRIT(DD3F)E ROUTINE TO WRITE A MAGTAPE BUFFER.

MTXBYT(00B4)V EXTRA "BYTE" COUNT FOR MARKING FILES.

MULCOD(000E)C CODE FOR MUL TOKEN.

MULPNT(00C6)V POINTER USED IN FP OUTPUT CONVERSION TO ACCESS PROPER POWER OF 10.

N Symbols

N1 (05E6)V ASCII # LSD.

N11 (05F0)V ASCII # DIGIT.

N12 (05F1)V ASCII # MSD.

NE (E9EA)E ENTRY FOR FUZZY NE COMPARE.

NECOD (0012)C CODE FOR NE TOKEN.

NEWBFR (F031)E ROUTINE TO GET A NEW INPUT BUFFER.

NEXCOD (005D)C CODE FOR NEX TOKEN.

NEXT (A066)E ROUTINE TO IMPLEMENT NEXT COMMAND.

NGTAB (BA3A)C FP TABLE OF NEGATIVE POWERS OF 10.

NIODEV (C42C)E ROUTINE TO SET NO I/O DEVICE ERROR MESSAGE.

NLPTR (004F)V NEXT LINE POINTER - THE ADDRESS OF THE NEXT LINE TO BE EXECUTED.

NMISRV (FDBC)E NON MASKABLE INTERRUPTE (NMI) SERVICE ROUTINE FOR THE MAGTAPE.

NOCASE (C24C)E NO CASE - ROUTINE TO IMPLEMENT SET NOCASE COMMAND.

NOCCOD (009D)C CODE FOR NOC TOKEN.

NOKCOD (009B)C CODE FOR NOK TOKEN.

NOKEY (C244)E NO KEY - ROUTINE TO IMPLEMENT SET NOKEY COMMAND.

NOOUT (0087)V OUTPUT BUFFER STATUS FLAG BYTE.

NOP (D158)E NO OPERATION - ROUTINE TO IMPLEMENT NO OPERATION TOKENS.

NORCOD (0099)C CODE FOR NOR TOKEN.

NORM (B343)E ROUTINE TO NORMALIZE FP NUMBER AND CHECK FOR OVER/UNDERFLOW OR ZERO RESULT.

NORMAL (C239)E NORMAL - ROUTINE TO IMPLEMENT SET NORMAL COMMAND.

NORMR (B359)E ALTERNATE ENTRY TO NORM VIA PSHRET.

NOT (EE0E)E LOGICAL NOT SUBROUTINE.

NOTCOD (0023)C CODE FOR NOT TOKEN.

NS (05F2)V ASCII # SIGN.

NTPTR (0051)V NEXT TOKEN POINTER - THE ADDRESS OF THE NEXT TOKEN TO BE EXECUTED.

NULCHR(007A)V I/O SYSTEM NULL CHARACTER. THIS CHARACTER WILL BE DELETED FROM THE INPUT STEAM BEFORE BEING PASSED TO BASIC.

NULLTG(0000)C NULL TAG - STACK TAG FOR ONE NULL BYTE.

NUMCOD(0080)C CODE FOR NUM TOKEN.

NUMFLG(00F0)V NUMBER FLAG.

NUMOUT(9D43)E UNLEX ROUTINE TO CONVERT FPN TO ASCII AND OUTPUT IT.

0 Symbols

OF (9F87)E ROUTINE TO IMPLEMENT GOTO EXP OF LIST AND GOSUB EXP OF LIST COMMANDS.

OFCOD (008B)C CODE FOR OF TOKEN.

OFF (C107)E ROUTINE TO IMPLEMENT OFF COMMAND.

OFFCOD(00A8)C CODE FOR OFF TOKEN.

OLD (04A1)C OLD COMMAND I/O SYSTEM CONTROL CONSTANT.

OLDCOD(006E)C CODE FOR OLD TOKEN.

OLDONE(E08C)E SPECIAL ENTRYPOINT FOR OLD CONTROL.

ON (C11B)E ROUTINE TO IMPLEMENT ON COND THEN LINE COMMAND.

ONCOD (00A9)C CODE FOR ON TOKEN.

ONTBL (03B9)V ON TABLE - THE ADDRESSES OF THE ON UNIT PROCESSING COMMANDS.

ONUNIT(C100)E ROUTINE TO IMPLEMENT ON COND AND OFF COND COMMANDS.

OPECOD(0070)C CODE FOR OPE TOKEN.

OPEN (0300)C OPEN COMMAND I/O SYSTEM CONTROL CONSTANT.

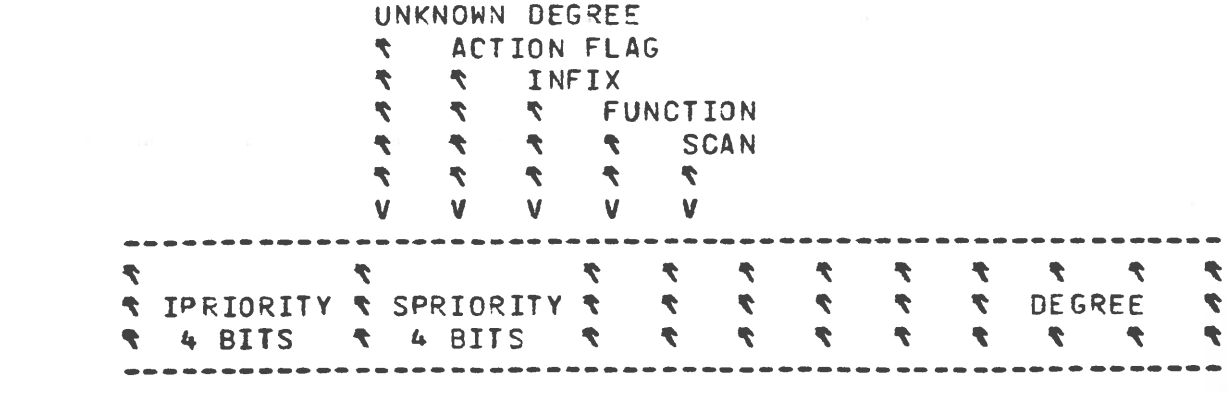
OPRADR(0056)V OPERATOR ADDRESS - HOLDING AREA FOR PRIMARY EVALUATOR DISPATCH DATA.

OPRCL (AAE4)E OPERATOR CALL - ROUTINE TO DO BANK SWITCHED LINKAGES FOR THE EVALUATOR.

OPRRTN(0437)V OPERATOR RETURN - SAVE AREA FOR RETURN POINT FROM OPRCL

OPRTBL(BAA2)C OPERATOR TABLE - SECTION OF EVALUATOR DISPATCH TABLES.

OPTABL(8F48)C OPTABLE, TRANSLATOR OPERATOR TABLE. THIS TABLE HAS ONE ENTRY FOR EACH TOKEN. EACH ENTY HAS THE FOLLOWING FORMAT.



OR (EE25)E LOGICAL OR SUBROUTINE.

ORCOD (000B)C TOKEN CODE FOR LOGICAL OR.

OUTBFR (C374)E ROUTINE TO SEND A FULL OUTPUT BUFFER TO THE CURRENT I/O DEVICE.

OVLN2 (E936)C FLOATING POINT CONSTANT = LOG2(E).

OVLTN (E946)C FP CONSTANT = LOG10(E).

P Symbols

PAETG (000B)C POINTER TO ARRAY ELEMENT TAG - STACK TAG.

PAGCOD(007B)C CODE FOR PAG TOKEN.

PAGE (1620)C PAGE COMMAND I/O SYSTEM CONTROL CONSTANT.

PARCNT(00E8)V PARENTHSIS COUNT.

PARSE (A206)E PARSER ENTRY POINT.

PCHAR (CBEE)E ROUTINE TO SEND A SINGLE CHARACTER TO THE DISPLAY. NOTE: DOES NOT LOOK AT PAGE FULL STATUS.

PFLG (00F2)V PLUS FLAG.

PGCTR (058C)V AUTO-ERASE DELAY COUNTER.

PGFVCT(0003)C BANK CONTROL CONSTANT FOR PAGE FULL.

PGMEVL(9683)E PROGRAM EVALUATOR - ROUTINE TO START AND RUN A PROGRAM.

PGMLIS(913E)E PROGRAM LIST BUILDER, INSERTS A NEW LINE INTO THE APPROPRIATE PLACE IN THE PROGRAM.

PGMPTR(003D)V PROGRAM POINTER - THE ADDRESS OF THE FIRST PROGRAM LINE.

PGMTG (0006)C PROGRAM TAG - STACK TAG FOR A POINTER TO A PPOGRAM LINE.

PGNEND(C600)C PAGE N END - THE ADDRESS OF THE FIRST FREE BYTE IN USER RAM. THE END OF SYSTEM RAM+1.

PGZEND(00F8)C PAGE ZERO END - THE FIRST FREE BYTE IN PAGE ZERO.

PI (E918)E ROUTINE TO STACK PI.

PIAEND(058A)C PIA END - THE ADDRESS OF THE END OF THE PIA TABLE.

PIAEOI(8780)C THE ADDRESS OF THE FIRST IEC BUS PIA.

PIAHX (8794)C THE ADDRESS OF THE CRT HI X DAC.

PIAHY (878C)C THE ADDRESS OF THE CRT HI Y DAC.

PIAKB (87A8)C THE ADDRESS OF THE KEYBOARD INTERFACE PIA.

PIALT (87AA)C THE ADDRESS OF THE PIA FOR THE LIGHTS AND THE REST OF THE KEYBOARD.

PIALX (8796)C THE ADDRESS OF THE CRT LOW X DAC.

PIALY (878E)C THE ADDRESS OF THE CRT LOW Y DAC.

PIAMTA(8798)C THE ADDRESS OF THE FIRST MAG TAPE PIA.

PIAMTB(879A)C THE ADDRESS OF THE SECOND MAG TAPE PIA.

PIASRQ(87B2)C THE ADDRESS OF THE SECOND IEC BUS PIA.

PIATBL(050C)C PIA TABLE - THE ADDRESS OF THE ORIGIN OF THE PIA TABLE.

PICOD (009F)C CODE FOR PI TOKEN.

PICON (E91E)C FP CONSTANT PI=3.1415926535...

PLOSTG(0001)C POINTER TO A LITERAL IN AN OBJECT STRING TAG - STACK TAG.

PLOTVL(93AA)E ROUTINE TO CONVERT USER SPACE TO SCREEN SPACE AND DUMP THE RESULTANT VECTOR INTO THE OUTPUT BUFFER.

PLSCOD(000C)C CODE FOR PLS TOKEN.

PLTAB (B9D2)C FP TABLE OF POSITIVE POWERS OF 10.

PNDEOF(006E)V PENDING EOF - IF NONE ZERO. THE NUMBER OF AN I/O UNIT WITH THE EOF CONDITION RAISED. NOTE PNDEOF=128 IF THE UNIT IS 0, THE INTERNAL TAPE.

PNDFLG(006D)V PENDING FLAGS - BIT FLAGS FOR THE ON CONDITIONS THE SYSTEM WILL ACCEPT. THE HIGH ORDER BIT IS SET BY BREAK.

PNTNTG(000A)C POINTER TO NAME TABLE NUMERIC TAG - STACK TAG.

PNTSTG(0008)C POINTER TO NAME TABLE STRING TAG - STACK TAG.

POINT (009C)V I/O SYSTEM POINTER TO PRESENT VALUE OR STRING.

POINTB(0476)V I/O PROCESSOR POINTER FOR RECOVERING AFTER PAGE FULL.

POLCOD(0064)C CODE FOR POL TOKEN.

POLL (F6A2)E ROUTINE TO HANDLE THE POLL STATEMENT.

POPES (D07C)E POP EVALUATOR STATUS - ROUTINE TO PULL EVALUATOR STATUS AND RESET IT IN PAGE ZERO.

POS (0070)C BANK SWITCH CONTROL CONSTANT.

POSCOD(0007)C CODE FOR POS TOKEN.

PPEOF (0465)V <PERCENTSIGN> MODE EOF CHARACTER.

PPEOR (0464)V <PERCENTSIGN> MODE EOL CHARACTER.

PPMODE(0077)V <PERCENTSIGN> MODE ENVOKED FLAG.

PPNUL (0466)V <PERCENTSIGN> MODE NULL CHARACTER.

PRGBLN(0A3E)C POINTER TO BLANK PROGRAM MAGTAPE HEADER.

PRGDAT(0A52)C POINTER TO "DATA" PROGRAM MAGTAPE HEADER.

PRGPGM(0A48)C POINTER TO "PROGRAM" PROGRAM MAGTAPE HEADER.

PRICOD(005C)C CODE FOR PRI TOKEN.

PRIDFT(C288)E ASCII OUTPUT CONTROL ROUTINE.

PRINT (0C20)C PRINT COMMAND I/O SYSTEM CONTROL FUNCTION.

PRINTF(0178)M ASCII FORMATTING MODULE.

PRISTG(C264)E ROUTINE TO PRINT A STRING.

PRIVAL(C27E)E ROUTINE TO PRINT A VALUE.

PRMCOD(00B4)C CODE FOR PRM TOKEN.

PRTERR(8338)E PRINT ERROR - THE ERROR MESSAGE WRITER.

PRTTG (0016)C PRINT TAG - STACK TAG FOR INFORMATION SAVED DURING PAGE FULL ON UNIT PROCESSING.

PSCTG (0009)C POINTER TO STRING COUNT STACK TAG.

PSHFPN(B6EB)E ROUTINE TO PUSH FP NUMBER ON STACK.

PSHRET(A9E7)E PUSH RETURN - SAVES RETURN ADDRESS ON A SECOND STACK.

PT   (AF7F)E ROUTINE TO HANDLE <PERCENTSIGN> CALL.

PTCOD (004D)C CODE FOR PT TOKEN.

PTRNTN(BE99)E ROUTINE TO STACK NUMERIC VARIABLE.

PTRNTS(BEB1)E ROUTINE TO STACK STRING VARIABLE.

PULFPN(B70F)E ROUTINE TO POP FP NUMBER FROM STACK.

PULLRN(AA19)E PULL REGISTER N - ROUTINE TO PULL ANY REGISTER FROM THE STACK.

PUPTAP(FB90)E ROUTINE TO INITIALIZE MAGTAPE HARDWARE.

PUSHES(D062)E PUSH EVALUATOR STATUS - ROUTINE TO SAVE EVALUATOR STATUS ON THE STACK AND TO CLEAR THE COPY IN PAGE ZERO.

PUSHX (AA08)E PUSH X - ROUTINE TO PUSH X AND GIVE IT ITM1TG.

PUSHXT(AA0A)E PUSH X - ROUTINE TO PUSH X AND GIVE IT A SPECIAL TAG.

PUTBYT(C33D)E ROUTINE TO PUT A CHARACTER IN THE OUTPUT BUFFER.

PUTCHR(9C6E)E ROUTINE TO PUT CHARACTERS INTO OUTPUT BUFFER.

PWRUP (BC4B)E POWER UP - THE SYSTEM POWER UP ROUTINE.

Q Symbols

Q0	(8B7C)C	POINTER TO THE BEGINNING OF THE KEYWORD TABLE.
Q1	(8BE7)C	POINTER TO THE BEGINNING OF THE 2ND QUARTER OF THE KEYWORD TABLE.
Q2	(8CA2)C	POINTER TO THE BEGINNING OF THE 2ND HALF OF THE KEYWORD TABLE.
Q3	(8D8E)C	POINTER TO THE BEGINNING OF THE 4RD QUARTER OF THE KEYWORD TABLE.
QACS	(95DE)E	ROUTINE TO IMPLEMENT THE ACS FUNCTION.
QASC	(8ED4)E	ROUTINE TO IMPLEMENT THE ASC STRING FUNCTION.
QASN	(95DE)E	ROUTINE TO IMPLEMENT THE ASN FUNCTION.
QATN	(95FE)E	ROUTINE TO IMPLEMENT THE ATN FUNCTION.
QAXIS	(884C)E	ROUTINE TO IMPLEMENT THE AXIS COMMAND.
QCAT	(8D4E)E	ROUTINE TO IMPLEMENT THE CAT () STRING FUNCTION.
QCHR	(8EEC)E	ROUTINE TO IMPLEMENT THE CHR STRING FUNCTION.
QCROSS	(8F1A)E	ROUTINE TO IMPLEMENT THE POINTER COMMAND.
QDRAW	(9215)E	ROUTINE TO IMPLEMENT THE DRAW COMMAND.
QEND	(C11E)C	LAST POSITION IN THE TYPE-AHEAD QUE.
QGIN	(9231)E	ROUTINE TO IMPLEMENT THE GIN COMMAND.
QIN	(0072)V	TYPE-AHEAD QUE INPUT POINTER.
QLEN	(8EBC)E	ROUTINE TO IMPLEMENT THE LEN STRING FUNCTION.
QMOVE	(9223)E	ROUTINE TO IMPLEMENT THE MOVE COMMAND.
QOUT	(0074)V	TYPE-AHEAD QUE OUTPUT POINTER.
QPOS	(8BE2)E	ROUTINE TO IMPLEMENT THE POS STRING FUNCTION.
QRDRAW	(921C)E	ROUTINE TO IMPLEMENT THE RDRAW COMMAND.
QREP	(8DC1)E	ROUTINE TO IMPLEMENT THE REP STRING FUNCTION.
QRMOVE	(922A)E	ROUTINE TO IMPLEMENT THE RMOVE COMMAND.
QRND	(96F9)E	ROUTINE TO IMPLEMENT THE RND FUNCTION.
QROTAT	(9055)E	ROUTINE TO IMPLEMENT THE ROTATE COMMAND.

```
QSCALE(9176)E    ROUTINE TO IMPLEMENT THE SCALE COMMAND.
QSEG  (8C8A)E    ROUTINE TO IMPLEMENT THE SEG STRING FUNCTION.
QSETFU(97A2)E    ROUTINE TO IMPLEMENT THE FUZZ COMMAND.
QSTR  (8BA6)E    ROUTINE TO IMPLEMENT THE STR STRING FUNCTION.
QTRNSL(C62A)C    THE ADDRESS IN TRANSL THAT IS STACKED FOR EVAL.
QUSRFN(9380)C    THE ADDRESS IN EVLEN STACKED FOR A FNX CALL.
QVAL  (8B77)E    ROUTINE TO IMPLEMENT THE VAL STRING FUNCTION.
QVIEW (9130)E    ROUTINE TO IMPLEMENT THE VEIWPORT COMMAND.
QWINDO(90CC)E    ROUTINE TO IMPLEMENT THE WINDOW COMMAND.
```

R Symbols

R0	(0000)V	THE FIRST OF 24 16 BIT SYSTEM REGISTERS.
R1	(0002)V	SYSTEM REGISTER N.
R2	(0004)V	" "
R3	(0006)V	" "
R4	(0008)V	" "
R5	(000A)V	" "
R6	(000C)V	" "
R7	(000E)V	" "
R8	(0010)V	" "
R9	(0012)V	" "
R10	(0014)V	" "
R11	(0016)V	" "
R12	(0018)V	" "
R13	(001A)V	" "
R14	(001C)V	" "
R15	(001E)V	" "
R16	(0020)V	" "
R17	(0022)V	" "
R18	(0024)V	" "
R19	(0026)V	" "
R20	(0028)V	" "
R21	(002A)V	" "
R22	(002C)V	" "
R23	(002E)V	" "
RAD	(C221)E	ROUTINE TO IMPLEMENT SET RAD COMMAND.
RADCOD	(0097)C	CODE FOR RAD TOKEN.

RADCON(E9DA)C CONSTANT FOR RANGE REDUCTION IN TRIG ROUTINES. 4/PI
RADCON+8 = FP CONSTANT PI/4

RBYCOD(0077)C CODE FOR RBY TOKEN.

RBYTE (F18F)E ROUTINE TO IMPLEMENT RBYTE.

RBYTVL(F20F)E ROUTINE TO GET A SINGLE BYTE FROM THE GPIB IN
RBYTE FORMAT.

RDIG (0602)V RIGHT DIGITS PRINTING COUNT.

RDIGA (0603)V RIGHT PRINT POINTER (ADDRESS).

RDRAW (0030)C BANK SWITCH CONTROL CONSTANT.

RDRCOD(0004)C CODE FOR RDR TOKEN.

REACOD(0067)C CODE FOR REA TOKEN.

READ (0EA2)C READ COMMAND I/O SYSTEM CONTROL CONSTANT.

READRS(FD54)E MAGTAPE DRIVER TO READ A PHYSICAL RECORD.

REASTG(CCCF)E ROUTINE TO READ A STRING.

REAVAL(CCAA)E ROUTINE TO READ A VALUE.

RECONT(0085)V PRESENT AVAILABLE RECORD COUNT FOR OPEN
MAGTAPE FILE.

RECNO (007E)V FILE SYSTEM VARIABLE, PRESENT FILE REQUESTED.

RELBL (0B0C)C RELABEL COMMAND I/O SYSTEM CONTROL CONSTANT.

RELCOD(008A)C CODE FOR REL TOKEN.

RELTBL(A73C)C RELABEL TABLE IN PARSE.

REMCOD(00A4)C CODE FOR REM TOKEN.

RENCOD(0057)C CODE FOR REN TOKEN.

RENUM (9DC8)E RENUMBER ACCORDING TO REGISTER DIRECTIVES.

RENUMB(9D5A)E RENUMBER ACCORDING TO STACK DIRECTIVES.

REP (0078)C BANK SWITCH CONTROL CONSTANT.

REPCOD(0006)C CODE FOR REP TOKEN.

RESCOD(0056)C CODE FOR RES TOKEN.

```
RESTS  (BF65)E    ROUTINE TO IMPLEMENT RESTORE STATEMENT NUMBER.
RESTZ  (BF5C)E    INTERNAL ROUTINE TO RESET DATA STATEMENT POINTERS.
RETCOD (005E)C    CODE FOR RET TOKEN.
RETH   (0024)V    TEMPORARY FOR RETURN ADDRESS
RETL   (0025)V    " "
RETURN (9FE4)E    ROUTINE TO IMPLEMENT RETURN STATEMENT.
REWIND (FBD7)E    ROUTINE TO REWIND THE MAGTAPE.
RMOCOD (0002)C    CODE FOR RMO TOKEN.
RMOVE  (0040)C    BANK SWITCH CONTROL CONSTANT.
RND    (0088)C    BANK SWITCH CONTROL CONSTANT.
RNDATA (B949)E    ROUTINE TO ROUND FP NUMBER AND CONVERT TO ASCII.
RNDCOD (C042)C    CODE FOR RND TOKEN.
ROTATE (C020)C    BANK SWITCH CONTROL CONSTANT.
ROTCOD (0082)C    CODE FOR ROT TOKEN.
ROW    (0595)V    KEYBOARD DRIVER ROW OF PRESENT KEY.
ROWCT  (00AD)V    DISPLAY CHARACTER ROW COUNTER.
RPN    (E354)E    RIGHT PARENTHESIS TOKEN HANDLER.
RPNCOD (008F)C    CODE FOR RPN TOKEN.
RTRN   (A9FD)E    ROUTINE TO PULL USE ALTERNATE RETURN POINT.  SEE
                  PSHRET.
RTRNTG (0015)C    RETURN TAG - STACK TAG FOR A INTERNAL RETURN ADDRESS.
RUN    (9F16)E    ROUTINE TO IMPLEMENT RUN COMMAND.
RUNBNK (AA7F)E    ROUTINE TO RUN ALL BANKS FOR POWER UP ECT..
RUNCOD (0055)C    CODE FOR RUN TOKEN.
RWNDFG (0010)C    DSPSTT FLAG BIT -- REWIND PENDING.
```

S Symbols

SAFE (A99E)E ROUTINE TO CALL IF SYSTEM IS SAFE FOR ROM INTERRUPTS BUT IT IS DISABLED.

SAVCOD(C05B)C CODE FOR SAV TOKEN.

SAVE (0121)C SAVE COMMAND I/O SYSTEM CONTROL CONSTANT.

SBP (0043)V STACK BASE POINTER - BOTTOM OF EXECUTION STACK.

SCACOD(0081)C CODE FOR SCA TOKEN.

SCALE (0008)C BANK SWITCH CONTROL CONSTANT.

SCLMAT(94F4)E SCALER-MATRIX - ROUTINE TO APPLY A SCALER OPERATOR OVER A SCALER AND AN ARRAY.

SCMSED(0479)V PRESENT SCRAMBLE CHARACTER.

SCORE (0038)V USED IN ROUTINE UNCOMP TO DECIDE WHEN TO UNSTACK AND OPERATOR INTO THE RESULT.

SCRTCH(0235)V SCRATCH - SYSTEM WORK AREA IN PAGE N.

SECCOD(0072)C CODE FOR SEC TOKEN.

SECFNC(1025)C SECRET COMMAND I/O SYSTEM CONTROL CONSTANT.

SEG (0080)C BANK SWITCH CONTROL CONSTANT.

SEGCOD(C095)C CODE FOR SEG TOKEN.

SEMCOD(008E)C CODE FOR SEM TOKEN.

SEMITG(0011)C SEMI COLON TAG - STACK TAG.

SERCHS(FE0C)E MAGTAPE HARDWARE DRIVER TO DO FAST SEARCH.

SETARG(CE7C)E SET UP FOR TYPARG - ROUTINE TO PRIME TYPARG CALLS.

SETBNK(A9D5)E SET BANK - ROUTINE TO SWITCH BANK AND DO HOUSEKEEPING.

SETCOD(0051)C CODE FOR SET TOKEN.

SETERA(D16A)E SET ERROR AND TAKE EXIT A - ROUTINE TO SET THE ERROR CODE AND RETURN USING DREXTA.

SETERR(D155)E SET ERROR CODE - ROUTINE TO SET THE ERROR CODE AND RETURN TO THE ORIGINAL CALLER.

SETFUZ(00C0)C BANK SWITCH CONTROL CONSTANT.

SETRND(EFA3)E SET DEFAULT VALUE FOR KERNAL OF RND.

IFT1(0593)V KEYBOARD DRIVER SHIFT KEY FLAG.

SHIFT2(0594)V ANOTHER KEYBOARD DRIVER SHIFT KEY FLAG.

SHMR (B21D)E ROUTINE TO RIGHT SHIFT FRACTION PRIOR TO ADD/SUBTRACT.

SHNTCT(0033)V VARIABLE USED TO RETURN THE NUMBER OF BYTES NEEDED TO STORE THE POSTFIX LINE. GENERATED BY SHUNT, RETURNED TO TRANSL.

SHUNT (A0DE)E CONVERTS THE PREFIX-INFIX INTERNAL NOTATION LINES PRODUCED BY LEX INTO POSTFIX, EXECUTABLE LINES.

SIG (EDE7)E CODE TO COMPUTE SIGN OF A FP NUMBER.

SIGCOD(0041)C CODE FOR SIG TOKEN.

SIGNS (00C3)V FLAG SET IN FP INPUT CONVERSION:"SIGN OF NUMBER."

SINCOD(004C)C CODE FOR SIN TOKEN.

SINTHA(04DA)V SINE OF ROTATED ANGLE FOR RELATIVE GRAPHICS. AN FPN.

SIZCOD(00AA)C CODE FOR SIZ TOKEN.

IZERR(0006)C SIZE ERROR - BREAK POINT FROM SIZE ERRORS TO NORMAL ONES.

SKIPS (FE84)E MAGTAPE DRIVER ROUTINE TO SKIP A RECORD.

SLN (EBB6)E ROUTINE TO LEFT SHIFT FRACTION OF FP NUMBER.

SLOPH (0000)C SLOP HIGH BYTE - SPACE DIFFERENCE BETWEEN STACK REQUIPMENT AND FUDGE.

SLOPL (0064)C SLOPE LOW BYTE.

SM2COD(00AF)C CODE FOR SM2 TOKEN.

SNDBFR(C404)E ROUTINE TO SEND AN OUTPUT BUFFER AND SET EOI BIT.

SNGMAT(94CA)E SINGLE MATRIX - ROUTINE TO APPLY A MONADIC SCALER OPERATOR OVER ONE ARRAY.

SP1 (00DB)V RELATIVE STRING POINTER 1.

SP2 (00DE)V RELATIVE STRING POINTER 2.

SP3 (00E1)V RELATIVE STRING POINTER 3.

SP4 (00E4)V RELATIVE STRING POINTER 4.

PACE (C1D5)E ROUTINE TO IMPLEMENT SPACE COMMAND.

SPACOD(00A0)C CODE FOR SPA TOKEN.

```
SPCIOB(ABB3)E    ENTRY POINT INTO IOSCAN TO SCAN A RBYTE/WBYTE LIST.

SPCIOF(1020)C    SPECIAL FUNCTION COMMAND I/O SYSTEM CONTROL CONSTANT.
                 THIS IS USED BY SOME INTERNAL BUFFER REQUSETS.

SPOLON(F798)E    ROUTINE TO SEND SERIAL POLL ENABLE.

SQR   (EB0F)E    ROUTINE TO COMPUTE SQUARE ROOT OF FP NUMBER.

SQRCOD(0040)C    CODE FOR SQR TOKEN.

SRQCOD(00AC)C    CODE FOR SRQ TOKEN.

SRQOFF(C19F)E    RESETS SRQ PENDING BIT.

SRQRDY(F5AC)E    ROUTINE TO PROCESS SRQ LEVEL REQUESTS.

STAT37(007B)V    4051 PROCESSOR IN SECRET OUTPUT STATUS.

STAX  (0457)E    SELF MODIFYING CODE TO STORE ACC=B AT (ACC-A)+(X).
                 STAX: STA A STAX+4  :MODIFY DISP FIELD
                  STA B 0,X  :STORE DATA
                  RTS

STECOD(0088)C    CODE FOR STE TOKEN.

STGCOD(00B1)C    CODE FOR STG TOKEN.

STKBLD(E82E)E    ROUTINE TO BUILD A BACKWARDS STACK FOR I/O LISTS.

STKUSE(0597)V    KEYBOARD DRIVER STACK USAGE COUNTER.

STOCOD(005F)C    CODE FOR STO TOKEN.

STOP  (9F4E)E    ROUTINE TO IMPLEMENT STOP COMMAND.

STPTR (003B)V    SYMBOL TABLE POINTER.

STR   (0098)C    BANK SWITCH CONTROL CONSTANT.

STRCOD(001D)C    CODE FOR STR TOKEN.

STRING(0010)M    MODULE WITH THE STRING FUNCTIONS IN IT.

SUM   (9620)E    ROUTINE TO IMPLEMENT THE SUM FUNCTION.

SUMCOD(0019)C    CODE FOR SUM TOKEN.

SWIERR(BC2E)E    SWI ERROR - ROUTINE TO HANDLE SWI INTERRUPTS.

SYMTAB(B0D6)E    SYMBOL TABLE ROUTINE, BUILDS AND SEARCHES THE SYMBOL
                 TABLE.

SYSERR(BC30)E    SYSTEM ERROR - ROUTINE TO HANDLE THINGS THAT CAN NOT
                 OCCUR.
```

```
T1      (00BF)V     T1 - T4 ARE FP INTERNAL SCRATCH BYTES.
T2      (00C0)V     "
T3      (00C1)V     "
T4      (00C2)V     "

TABAD  (E94E)C      POINTER TO TABLE OF POWERS OF (2**(1/16))**M.
TABCNT (060B)V      CURRENT LOGICAL OUTPUT POSITION.
TABPNT (00C4)V      POINTER USED FOR VARIOUS FP OPERATIONS.
TABPTR (00A0)V      IOSCAN NAME TABLE POINTER.
TAGPTR (0088)V      IOSCAN STACK POINTER.
TAN15  (E9B2)C      FP CONSTANT = TAN(15 DEGREES).
TANCOD (004A)C      TOKEN CODE FOR TAN.
TAPE   (09C0)E      ROUTINE TO PROCESS MARK, FIND COMMANDS.
TAPF2  (E12F)E      ROUTINE TO PROCESS LIST, SAVE, OLD COMMANDS.
TAPFIL (E129)E      SIMIALAR TO TAPF2.
TCOL   (00A2)V      TEMPORARY MATRIX COLUMN COUNTER FOR I/O.
TEMPX  (04F2)V      TEMP. STORAGE OF FPN FOR GRAPHICS.
TEMPY  (0502)V      TEMP. STORAGE OF FPN FOR GRAPHICS.
TESTCS (AA52)E      DOES A CONDITIONAL UPCASE DEPENDING ON SET CASE/NOCASE.
TFLGS1 (0035)V      TRANSLATOR FLAGS BYTE.
                    -------------------------------------------------
                    ↑ IMM ↑ ASS ↑ DEF ↑ STR ↑ PARM ↑ RAND ↑ OF ↑     ↑
                    ↑     ↑     ↑ FLG ↑     ↑      ↑      ↑    ↑     ↑
                    -------------------------------------------------
THECOD (0089)C      CODE FOR THE TOKEN.
TLICOD (007A)C      CODE FOR TLI TOKEN.
TLIST  (E207)E      ROUTINE TO PROCESS THE TLIST COMMAND.
TMP1   (00A5)V      TEMP. STORAGE FOR THE DISPLAY DRIVER. VECTOR INFORMATION
TMP2   (00A6)V      SIMILAR TO TMP1.
```

```
TMP3   (00A7)V    SIMILAR TO TMP1.
TMPHX  (00AA)V    CURRENT POSITION OF X AXIS DISPLAY IN TEKPOINTS.
TMPHY  (00A8)V    CURRENT POSITION OF Y AXIS DISPLAY IN TEKPOINTS.
TMPLX  (00AB)V    THE LOWER 8 BITS OF TMPHX.
TMPLY  (00A9)V    THE LOWER 8 BITS OF TMPHY.
TO2COD (0087)C    CODE FOR TO2 TOKEN.
TOCOD  (0086)C    CODE FOR TO TOKEN.
TPOINT (001C)V    TABLE POINTER.
TRACE  (C232)E    ROUTINE TO IMPLEMENT SET TRACE COMMAND.
TRACOD (0098)C    CODE FOR TRA TOKEN.
TRANSL (A8B0)E    TRANSLATOR MAINLINE.
TRAPS  (FFF8)C    BASE ADDRESS FOR SYSTEM TRAPS.
TRASH  (011E)V    I/O SYSTEM SCRATCH VARIABLE.
TRIG   (EA2C)E    ROUTINES TO COMPUTE SIN, COS, AND TAN.
TRN    (0021)C    BANK SWITCH CONTROL CONSTANT.
TRNCOD (001A)C    CODE FOR TRN TOKEN.
TRUTH  (EBCB)E    ROUTINE TO DETERMINE IF FP NUMBER IS CLOSER TO 0 OR 1.
TST8NF (FF8C)E    ROUTINE TO TEST FOR 8NFR'S (PHYSICAL END OF FILE).
TSTINT (93C4)E    TEST INTERRUPT - EVALUATOR ROUTINE TO TEST FOR RAISED
                  ON CONDITIONS AND PENDING USER KEYS.
TTY2   (0592)V    KEYBOARD TTY LOCK FLAG.
TYPARG (CE77)E    TYPE OF ARGUMENT - ROUTINE TO TEST AND SIMPLIFY FUNCTION
                  ARGUMENTS ON THE STACK.
TYPASC (DA2E)C    POINTER TO "ASCII" HEADER.
TYPBIN (DA26)C    POINTER TO "BINARY" HEADER.
TYPCOD (003E)C    CODE FOR TYP TOKEN.
TYPE   (CE00)E    ROUTINE TO PROCESS THE TYPE COMMAND.
```

TYPFIL(E874)E ENTRY CALL TO FILE SYSTEM FOR TYP N.

TYPIN (8033)E LINE EDITOR MAIN ENTRY POINT.

TYPLST(DA36)C POINTER TO "LAST" HEADER.

TYPNEW(DA1E)C POINTER TO "NEW" HEADER.

TYPRES(CF77)E TYPE OF RESULT - ROUTINE TO TEST FOR VALID RESULT AREAS FOR SYSTEM FUNCTIONS.

TYPSEC(DA5C)C POINTER TO "SECRET" HEADER.

UNADR (AF1D)E ROUTINE TO UNADDRESS THE I/O SYSTEM. RESETS DISPLAY AND I/O LIGHT AND I/O VARIABLES.

UNCOMP(98AA)E UNCOMPILER, CONVERTS POSTFIX INTERNAL NOTATION LINES TO PREFIX-INFIX INTERNAL NOTATION.

UNDER (B3BC)E SAME AS ZANS - PUSHES 0 ON STACK.

UNICOD(0075)C CODE FOR UNI TOKEN.

UNIT (080C)C UNIT COMMAND I/O SYSTEM CONTROL CONSTANT.

UNLEX (9B90)E UNLEXICALIZE, CONVERTS INTERNAL NOTATION INTO ASCII.

UP (EE6E)E ROUTINE TO COMPUTE FP A**B.

UPCASE(AA5A)E CONVERTS INPUT DATA TO UPPERCASE.

UPCOD (0010)C CODE FOR UP TOKEN.

UPDLEN(ADCE)E ROUTINE TO UPDATE A STRING LENGTH AFTER INPUT/READ.

USICOD(0091)C CODE FOR USI TOKEN.

USING (A84C)E ROUTINE TO SET UP POINTERS FOR PRINT USING COMMAND

USRORG(0047)V USER ORIGIN - THE ADDRESS OF THE FIRST FREE BYTE OF RAM. THIS VARIABLE SHOULD BE USED INSTEAD OF PGMEND BY SYSTEM MODULES.

UTILS (A994)M SYSTEM UTILITY ROUTINES.

V Symbols

VAL (0090)C	BANK SWITCH CONTROL CONSTANT.
VALCOD(001F)C	CODE FOR VAL TOKEN.
VALTG (000C)C	VALUE TAG - STACK TAG FOR A FLOATING POINT NUMBER.
VECTOR(CB81)E	ROUTINE TO DISPLAY VECTORS ON THE DISPALY.
VIECOD(006C)C	CODE FOR VIE TOKEN.
VIEW (001C)C	BANK SWITCH CONTROL CONSTANT.
VLDKEY(05A6)V	KEYBOARD HANDLER TEMPORARY KEY REGISTER.

W Symbols

```
WAICOD(007F)C    CODE FOR WAI TOKEN.
WAIT  (C250)E    ROUTINE TO IMPLEMENT WAIT COMMAND.
WAIT. (C6E6)E    ROUTINE TO WAIT 40MSEC.
WBYCOD(0078)C    CODE FOR WBY TOKEN.
WBYTE (F188)E    ROUTINE TO HANDLE WBYTE COMMAND.
WBYTVL(F1E2)E    ROUTINE TO SEND ONE BYTE FROM THE WBYTE LIST.
WINCOD(006B)C    CODE FOR WIN TOKEN.
WINDOW(0018)C    BANK SWITCH CONTROL CONSTANT.
WREND.(FE4E)E    ROUTINE TO FINISH WRITING A MAGTAPE RECORD.
WRICOD(0068)C    CODE FOR WRI TOKEN.
WRISTG(CD3D)E    ROUTINE TO WRITE A STRING.
WRITE (0F21)C    WRITE COMMAND I/O SYSTEM CONTROL CONSTANT.
WRIVAL(CD6F)E    ROUTINE TO WRITE A VALUE.
WRRS  (FD25)E    MAGTAPE DRIVER ROUTINE TO WRITE A PHYSICAL RECORD.
```

X Symbols

X0	(00D3)V	X0 ... X5 IS A SIX BYTE ARRAY FOR HOLDING FRACTIONS DURING FLOATING POINT OPERATIONS.
X1	(00D2)V	" "
X2	(00D1)V	" "
X3	(00D0)V	" "
X4	(00CF)V	" "
X5	(00CE)V	" "
XEQSP	(0049)V	EXECUTION STACK POINTER - SECOND STACK FOR RETURN POINTS.
XEQSTK	(0424)V	EXECUTION STACK SPACE - WORK AREA FOR EXECUTION STACK.
XFNBNK	(0469)V	EXTENDED FUNCTION BANKS - FOUR VARIABLES THAT HOLD THE BANK ADDRESSES OF THE EXTENDED FUNCTION ROMS.
XFNMAP	(8800)M	MODULE THAT HAS THE BANK HEADER FOR OVERFLOW ROM PACK.
XFRCTL	(C431)E	ROUTINE TO ATTACH AN APPROPRIATE I/O DRIVER GIVEN A.PRIM AND A TABLE POINTER. NOTE: THIS ROUTINE WILL ALSO ATTACH I/O DRIVERS FROM THE PIATBL.
XFRSCN	(E07C)E	MAGTAPE CONTROL ROUTINE TO INITIALIZE MAGTAPE OPERATIONAL REQUESTS: LIKE PRINT, OLD.
XLAST	(04C2)V	X AXIS LAST POSITION FPN IN USER UNITS.
XMAXW	(0482)V	PRESENT XMAX WINDOW (FPN).
XMINS	(04A2)V	PRESENT XMIN VIEWPORT (FPN).
XMINW	(047A)V	PRESENT XMIN WINDOW (FPN).
XNEW	(04EA)V	NEW POSITION FOR PLOTTING IN USER UNITS (FPN).
XSF	(04BA)V	PRESENT X AXIS SCALE FACTOR. XSF := (XMAX WINDOW - XMIN WINDOW) / (XMAX VIEWPORT - XMIN VIEWPORT)

Y Symbols

Y0	(00CD)V	Y0 ... Y5 IS A SIX BYTE REGISTER FOR HOLDING FRACTIONS DURING FLOATING POINT OPERATIONS.
Y1	(00CC)V	" "
Y2	(00CB)V	" "
Y3	(00CA)V	" "
Y4	(00C9)V	" "
Y5	(00C8)V	" "
YAXIS	(00A4)V	AXIS FLIP-FLOP FLAG.
YLAST	(04D2)V	SIMILAR TO XLAST.
YMAXW	(0492)V	SIMILAR TO XMAXW.
YMINS	(04B2)V	SIMILAR TO XMINS.
YMINW	(048A)V	SIMILAR TO XMINW.
YNEW	(04FA)V	SIMILAR TO XNEW.
YSF	(04CA)V	SIMILAR TO XSF.

Z Symbols

ZANS	(B3BC)E	ROUTINE TO PUT A FLOATING POINT 0.0 RESULT ON STACK.
ZDRAW	(1420)C	DRAW COMMAND I/O SYSTEM CONTROL CONSTANT.
ZERCNT	(0606)V	COMMON ZERO COUNT.
ZERO1	(0600)V	LEFT TRAILING ZERO COUNT.
ZERO2	(0601)V	RIGHT LEADING ZERO COUNT.
ZERO3	(0605)V	RIGHT TRAILING ZERO COUNT.
ZERSVH	(0607)V	ZERO SAVE ADDRESS, HIGH.
ZERSVL	(0608)V	ZERO SAVE ADDRESS, LOW.
ZGIN	(18A0)C	GIN COMMAND I/O SYSTEM CONTROL CONSTANT.
ZIX	(B3C2)E	ROUTINE TO OUTPUT A FP ZERO ON STACK.
ZIX4	(B3CA)E	ROUTINE TO ZERO TOP 4 WORDS OF STACK.
ZMOVE	(1520)C	MOVE COMMAND I/O SYSTEM CONTROL CONSTANT.
ZRDRAW	(9420)C	RDRAW COMMAND I/O SYSTEM CONTROL CONSTANT.
ZRMOVE	(9520)C	RMOVE COMMAND I/O SYSTEM CONTROL CONSTANT.
ZX	(0041)V	ZERO X - A VARIABLE CONTAINING ZERO THE CLEAR X WITH

TABLE OF CONTENTS

INTRODUCTION

This document provides information pertaining to the 4051 system firmware. The following is a list of the parts.

1) Routine documentation--this is a package containing documentation and interface specifications on routines within the 4051 which may be of use to potential ROM writers.

2) I/O System variable definitions--this is an appendix to the routine documentation. This appendix defines the I/O system variables which must be used for most of the I/O routines. This appendix is referred to in the documentation as the normal I/O system variables.

ABS

FUNCTION: ABSOLUTE VALUE

INPUT: FP NUMBER ON TOP OF STACK.

OUTPUT: FP NUMBER ON TOP OF STACK.

QACS/QASN/QATN

FUNCTION: COMPUTES INVERSE TRIG FUNCTIONS

INPUT: INPUT IS IN STACK

OUTPUT: ANSWER ON STACK.
ERRCD CONTAINS ERROR CODE.

USES: R1,R1+1
TABPNT

CALLS: PSHRET
DOFP
SQR
ZANS (IN NORM)
FPDIV
CMPFPN
FPNEG
PSHFPN
FPADD
A8X
FPMUL
RTRN

NOTES: ASIN AND ACOS ARE COMPUTED FROM

ASN=ATAN(X/SQR(1-X*X))

ATN IS COMPUTED BY REDUCING X TO ARCTAN (15 DEGREES) OR LESS AND THEN APPLYING A CONTINUED FRACTION APPROXIMATION.

ADRDEV

FUNCTION: THIS ROUTINE ASSIGNS PRIMARY AND SECONDARY ADDRESSES AND IN GENERAL SETS UP THE I/O SYSTEM FOR THE CURRENT I/O REQUESTED BY BASIC.

INPUTS: 1)OPRADR - 2 BYTES: A CONSTANT THAT IS SET UP BY THE EVALUATOR FOR EVERY I/O STATEMENT EXCEPT POLL, RBYTE AND WBYTE. THIS CONSTANT IS DIFFERENT FOR EVERY I/O STATEMENT AND ITS VALUE MAY BE FOUND IN IOKONS.

2)OPTIONAL ADDRESS DATA IS STORED ON THE STACK SOMEWHERE AND IS MARKED BY AN ATSNTG.

3)A,STAT - ATVLD BIT 2 <4,> IS SET IF OPTIONAL ADDRESS INFORMATION IS ON THE STACK.

4)PPMODE - FLAG FOR PERCENT MODE DELIMITERS.
PPNUL : PERCENT MODE NULL CHARACTER.
PPEOR : PERCENT MODE END-OF-LINE CHARACTER.
PPEOF : PERCENT MODE EOF CHARACTER.
IECRLF : OPTIONAL CR/LF DELIMITER MODE.

OUTPUTS : IOFUNC : SET TO CODE TO INDICATE PRESENT I/O FUNCTION.
A, STAT : SET APPROPRIATELY - SEE IOEQU FOR DEFINITION OF BITS.
ERRCD : SET IF ILLEGAL ADDRESS
A,PRIM : PRIMARY ADDRESS
A,SEC : SECONDARY ADDRESS
TABCNT =1 : COUNTER FOR PRINTF
IOSYSF =1 : I/O SYSTEM FLAG FOR FIRST ACCESS - GENERALLY; USED BY ROM PACKS (SPECIFICALLY FILE SYSTEM)
NULCHR : NULL CHARACTER - NORMALLY 255.
EOLCHR : END-OF-LINE CHARACTER NORMALLY <CR>
ETXCHR : EOF CHARACTER NORMALLY 255.
POINTR =65535 : FOR ON FULL
DR =65535 : FOR ON FULL
BLINK =0 : SO SCREEN I/O WILL WORK
NOOUT =0
CRSTAT =0 : INITIALIZE BUFFER FLAGS.

CALLS : ATPROC
MTPADR

NOTE : WILL ALSO TURN ON THE I/O LIGHT UNLESS PRIMARY ADDRESS IS 32, <DISPLAY> OR 34, <DATA>.

AFPITT

FUNCTION : CONVERTS AN ASCII STRING TO AN FPN.

INPUTS : R0 : POINTER TO FIRST CHARACTER OF STRING
R1 : POINTER TO DELIMITER OR ONE PAST LAST VALID CHARACTER.

OUTPUTS : FPC : RESULTANT NUMBER
R0 : POINTER TO DELIMITER USED
R1 : UN-MODIFIED

USES : POINT
CRSTAT

CALLS : PSHFPN
PULFPN
INPVAL

A(5-16)X AND D(5-13)X

FUNCTION : THESE ROUTINES ARE USED TO ADD OR SUBTRACT SMALL INTEGERS TO THE INDEX REGISTER. THE NUMBER IN THE NAME CALLED IS THE NUMBER ADDED TO IX.

INPUTS : X HAS ANY NUMBER.

OUTPUTS : IX IS ALTERED.

ASCFPN

FUNCTION : CONVERT AN INPUT STRING OF ASCII CHARACTERS INTO A FLOATING POINT NUMBER.
FORMAT IS:

(+/-)(DIGS)(.)(DIGS)(UPPER CASE E OR LOWER CASE E)(+/-)(DIGS)
OVERFLOW CAUSES LARGEST POSSIBLE NUMBER TO BE OUTPUT; NO ERROR OCCURS. UNDERFLOW CAUSES ZERO TO BE OUTPUT. IF TOO MANY DIGITS ARE SPECIFIED IN THE FRACTION THEY ARE IGNORED.

INPUTS : FIRST CHARACTER ASSUMED IN GLOBAL CHAR.
CALLS GETCHR TO PUT NEXT CHARACTER IN CHAR.

OUTPUTS : THE FLOATING POINT NUMBER IS PLACED ON THE STACK.
SETS SYSTEM FLAGS:

DIGFLG : 0 = DIGIT SEEN, <> 0 = NO DIGIT SEEN
ITSINT : 0 = INPUT NOT AN INTEGER <> 0 = INPUT A NON-NEGATIVE INTEGER
TFLGS1: SEE NOTES
EFLAG : <> 0 = ONLY AN E SEEN, 0 = ANYTHING ELSE
SIGNS : HIGH ORDER BIT = FRACTION NEGATIVE. LOW ORDER BIT = EXPONENT NEGATIVE.
ERRCD : IS CLEARED WHEN OUTPUT IS NORMAL.
EQU : <> 0 = NOT AN INTEGER

USES : CNT
X0...X5
T1...T4
DEXP,DEXP+1
PLTAB, THE TABLE OF POWERS OF 10
TABPNT

CALLS : GETCHR
PSHRET
TMULT
PSHFPN
NORMR (=PSHRET + NORM)
MAXANS (IN NORM)
ZANS (IN NORM)
FPDIV
FPMUL
A8X

NOTES: IF CNT=1 THEN IF NUMBER IS AN INTEGER THEN CLEAR BITS 2 AND 7 OF TFLGS1 ELSE SET EQU=1

BACKUP

FUNCTION : THIS ROUTINE USES R0 AS A PSEUDO STACK POINTER AND BACKS IT UP ONE STACK ENTRY. R0 HAS THE ADDRESS OF A VALUE TAG=1 FOR INPUT. IF THE BYTE IS NOT A LEGAL TAG SYSERR IS CALLED. IF THE TAG IS THE END OF STACK TAG R0 IS NOT ALTERED.

INPUTS : R0 IS THE ADDRESS OF A STACK TAG=1

OUTPUTS : R0 IS UPDATED TO REFLECT THE STACK ENTRY IS PASSED.

USES : R0

BELCAL

FUNCTION : BEEP THE 4051 BELL

BFRALC

FUNCTION : INITIALIZE BUFFER POINTERS BASED ON PRIMAPY ADDRESS.

INPUTS : A,PRIM : PRIMARY ADDRESS
MTSREG : TO DETERMINE IF 256 OR 128 BYTE MAGTAPE RECORDS.
A,STAT : TO DETERMINE IF INPUT OR OUTPUT.

OUTPUTS : A,STRT : POINTER TO FIRST SLOT
A,END = A,STRT=1
NOTE :A,STRT = 1 IS ALSO CLEARED, THIS IS NECESSARY FOR PUTBYT
A,MAX : POINTER TO LAST SLOT.

CALLS : LDXX

BININ

FUNCTION : ATTACHES PROPER I/O DEVICE FOR BINARY INPUT.

INPUT : NONE

OUTPUT : ERRCD SET IF ERROR IN HANDLER.

USES : NOTHING

CALLS : XFRCTL
DATIN
FILEIN
IECIN
MTPIN
NIOPEV

NOTES : ALL I/O SYSTEM VARIABLES MUST BE SET UP CORRECTLY.

BINOUT

FUNCTION : ATTACHES PROPER I/O DEVICE FOR BINARY OUTPUT.

INPUT : NONE

OUTPUT : ERRCD SET IF ERROR IN HANDLER.

USES : NOTHING

CALLS : XFRCTL
FILEOT
IECOUT
MTPOUT
NIOPEV

NOTES : ALL I/O SYSTEM VARIABLES MUST BE SET CORRECTLY.

CMPFPN

FUNCTION: COMPARES TWO FLOATING POINT NUMBERS.

INPUTS: A POSITIVE FLOATING POINT NUMBER (Y) POINTED TO BY TABPNT. A FLOATING POINT NUMBER (X) ON THE STACK. THE STACK CONTAINS TWO BYTES ON TOP FOLLOWED BY THE TAG FOR X AND THEN X ITSELF.

X AND Y ARE NOT CHANGED.

OUTPUTS: ACC A = 0 IF X = Y
= 1 IF X > Y
= -1 IF X < Y

COMPR

FUNCTION: COLLECTS ALL STILL VALID STRINGS, ARRAYS, AND PROGRAM LINES AS FAR DOWN (TOWARD APPR. 0) IN MEMORY AS POSSIBLE, OR TO SAY THAT ANOTHER WAY COLLECTS ALL AVAILABLE SPACE BETWEEN PGMEND AND LSP AND MOVES IT ABOVE LSP INTO THE FREE SPACE AREA.

INPUTS: ITEM 1 ON STACK >=0 AND <=32767. THIS IS THE NUMBER OF BYTES WHICH MUST BE RECOVERED FOR THE COMPRESS TO BE CONSIDERED SUCCESSFUL. IF LESS THAN THIS ARE RECOVERED AN ERROR - ERWSFL - IS GENERATED.

OUTPUTS: NONE (FREE SPACE\)

USES: R0,R2-R6

RESTRICTIONS, SUGGESTIONS, AND NOTES: 1) ALL ENTRIES ON THE STACK MUST BE CORRECTLY TAGGED WHEN THIS ROUTINE IS CALLED.

2) THIS ROUTINE DOES NOT EXPECT TO FIND (IE WILL NOT HANDLE CORRECTLY) A POINTER TO STRING COUNT - PSCTG - ON THE STACK.

CROS

FUNCTION : TO PLACE ON THE CRT SCREEN THE GRAPHIC POINTER IN RESPONSE TO THE IMPUT FROM THE JOY STICK AND TO TERMINATE THE PROCESS WHEN A LEGAL CHARACTER IS INPUT FROM THE KEYBOARD.

INPUTS : ANALOG VOLTAGE FROM JOY STICK AND TERMINATING CHARACTER FROM KEYBOARD.

OUTPUTS : THE VALUE OF THE CRT DA"S AT THE TIME A LEGAL KEYBOARD CHARACTER IS INPUT, IN TMPHY, TMPLY, TMPHX, TMPLX, AND THE KEYBOARD CHARACTER INPUT IN THE "A" REGISTER ON EXIT.

USES : NO PSEUDO REGISTERS

CALLS : GENCUR, KBOOUT

CRTRST

FUNCTION : RESET THE CRT PIA"S AND RESET THE OLD STATUS OF THE DISPLAY AS WELL AS SERVICING ANY PENDING PAGE, COPY, HOME, REWIND REQUESTS. GENERALLY CALLED TO RESTORE CRT OPERATION AFTER THE MAGTAPE HAS BEEN USED.

CALLS : DSPENL - SERVICES PENDING REQUESTS
ENABLE - TO OFFSET THE DSABLE OF THE MAGTAPE SECTION.

CTLCHR

FUNCTION : THIS ROUTINE SENDS CHARACTERS TO THE CRT BUT WILL WAIT UNTIL THE PAGE IS CLEARED IF THE DISPLAY IS IN FULL STATUS.

CTLCHR - SENDS ALL CHARACTERS TO THE SCREEN

INPUT : ACC, A - ASCII CHARACTER

CALLS : SETBNK
PCHAR
DSPCPY
FULSCN

NOTES : THIS ROUTINE DOES SOME SPECIAL HANDLING FOR THE OPTIONAL DISPLAY HANDLERS LIKE THE OPTION 1 INTERFACE THRU THE 3 BYTE DSPVCT VARIABLE. IT ALSO HANDLES THE SPECIAL PAGE FULL MODES AS WELL AS THE ON FULL CONDITION.

DEFPNT

FUNCTION: TO OUTPUT STRING OR NUMBERS ACCORDING TO THE DEFAULT IMAGE STRINGS CONTAINED WITHIN DEFPNT.

DEFAULT STRINGS ARE AS FOLLOWS:

1. STRINGS (DT=0)

FA

2. FOR NUMBERS (DT=1) IN THE RANGE OF 10E-3<|N|<10E-7 THE DEFAULT IMAGE STRING IS -

FD.FD

FOR ALL OTHER NUMBERS, THE DEFAULT IMAGE STRING IS -

FE

INPUTS: DT, DP, DI

OUTPUTS: OUTPUTS FORMATTED DATA THROUGH PUTBYT IN ACCORDANCE WITH ABOVE INPUTS.

USES: R0,R3

CALLS: UPCASE, PUTBYT

NOTES: THE CALLER OF DEFPNT MUST INSURE THAT THE INPUTS STAY VALID DURING THE DATA OUTPUT SEQUENCE.

DIM

FUNCTION : DIM IS CALLED AS A RESULT OF THE DIMENSION TOKEN. ITS FUNCTION IS TO CLEAR THE BRACKET COUNT AND SET THE DIMENSION FLAG IN LOCAL FLAG TO 1.

INPUTS : NONE

OUTPUTS : BRKCNT = 0
LCLFLG = LCLFLG + DIMFLG

USES : <-

CALLS : <-

NOTES : <-

DIMSTR

FUNCTION : DIMSTR IS USED TO DIMENSION A STRING VARIABLE.

INPUTS : INT1 MUST HAVE THE LENGTH TO BE DIMENSIONED, "X" REGISTER MUST HAVE VARIABLE NAME TABLE POINTER.

OUTPUTS : MEMORY IS ALLOCATED BY CREATING A NEW DATA ENTRY AS PER THE LENGTH VALUE. THE STRING PARAMETERS ARE SET IN THE NAME TABLE.

USES : R7

CALLS : COMPR, DISBLE, ENABLE

NOTES/ERRORS : PART OF THE ALLOCATION ROUTINE MAY INVOLVE CONSIDERABLE TIME SINCE THE MEMORY COMPRESS ROUTINE MAY BE INVOLKED. AN ERROR WILL RESULT IF THERE IS NOT ENOUGH MEMORY TO DO THE DIMENSIONING.

DISBLE

FUNCTION : THIS ROUTINE IS USED TO INCREMENT THE SYSTEM INTERRUPT MASK COUNTER TO DISABLE SYSTEM LEVEL INTERRUPTS. THIS IS USED TO PROTECT FUNCTIONS LIKE MEMORY COMPRESS FROM BEING INTERRUPTED FOR AN ABORT OPERATION. IF A FUNCTION IS LEAVING THE SYSTEM DISABLED FOR LONG PERIODS OF TIME THE ROUTINE SAFE SHOULD BE CALLED TO LET ROM PACKS PROCESS INTERRUPTS.

RESTRICTIONS, NOTES: SEE ALSO ENABLE AND SAFE.

DOFP

FUNCTION : FLOATING POINT INTERPRETER

INPUT : A TYPICAL CALL IS:

```
JSR DOFP
.BYTE FOP1 + FAR1
.BYTE FOP2 + FAR2
.WORD ....
.BYTE FRET
```

FOR EACH OPERATOR BYTE, DOFP PERFORMS FIRST THE 5-BIT OPERATION FOPX, THEN THE 3-BIT OPERATION FARX.. WORD"S OF DATA MAY BE PLACED AFTER SOME FOP"S. THE LAST FOP IS FRET. INDIVIDUAL FOP"S ARE DESCRIBED BELOW. THE FAR MAY BE:

FA (CALL FPADD)
FS (CALL FPSUB)
FM (CALL FPMUL)
FD (CALL FPDIV)

OUTPUT: THE STACK IS MANIPULATED ACCORDING TO THE FOP"S. CONTROL RETURNS TO THE BYTE AFTER FRET.

USES : R0,R0+1
T4
FPA
FPB
FPC

CALLS : PSHFPN
PULFPN
ABX
FPMUL
FPDIV
FPADD
FPSUB
JMPAX

FOP"S : 1. FHE: PUSH EXTENDED.

```
.BYTE FHEX + FAR
.WORD OPAD
```

THE FPN AT THE ADDRESS GIVEN IN THE .WORD (E.G.,OPAD) IS PUSHED ON THE STACK.

2. FHIN: PUSH INDIRECT
SAME AS FHEX EXCEPT OPAD CONTAINS THE ADDRESS OF THE FP NUMBER RATHER THAN THE NUMBER.

3. FRET: RETURN.

4. FLEX: PULL EXTENDED.

DOFP (continued)

SAME AS FHEX EXCEPT THE FP NUMBER IS POPPED FROM THE STACK TO OPAD.

5. FSWP: SWAP TWO STACK ENTRIES.
THE TOP TWO STACK ENTRIES ARE SWAPPED.

6. FLIN: PULL INDIRECT.
SAME AS FHIN EXCEPT THE FP NUMBER IS POPPED FROM THE STACK TO THE ADDRESS IN OPAD.

7. DUPLICATE STACK ENTRY.
THE TOP STACK ENTRY IS DUPLICATED.

8. FSAV: SAVE IN TEMPORARY.
THE TOP OF THE STACK IS POPPED TO THE GLOBAL FPA.

9. FRES: RESTORE FROM TEMPORARY.
THE FP NUMBER IN FPA IS COPIED TO THE TOP OF THE STACK.

10. FART: NO OP.
USED WHEN ONLY THE FAR IS TO BE PERFORMED.

11. FHIM: PUSH IMMEDIATE.
THE OPERAND IMMEDIATELY FOLLOWING IN THE .WORD IS TO BE PUSHED.

DRBUSY

FUNCTION : WAIT UNTIL DISPLAY IS IN A READY STATE. ALL COPIES, PAGES, VECTORS DONE.

NOTES : THIS ROUTINE MUST BE CALLED BEFORE ANY NEW COMMANDS ARE GIVEN TO THE DISPLAY. THIS IS NORMALLY DONE BY CRT SERVICE ROUTINES BUT CHECK THE ROUTINES BEFORE ASSUMING ANYTHING.

DSMOVE/DSDRAW/DSGRAF

FUNCTION: CONVERTS NUMBERS FROM GDU"S TO TEKPOINT"S (10 BIT DAC INFO.) AND THEN DISPLAYS A LIGHT OR DARK VECTOR TO THAT POINT.

INPUTS: TMP1 : 0 = DRAW <DSDRAW SETS IT>
1 = MOVE <DSMOVE SETS IT>
POINT : POINTS TO FLOATING POINT GDU VALUE.

USES : YAXIS = AXIS FLIP-FLOP CONTROL.
TMP2, TMP3
TMPHX, TMPLX
TMPHY, TMPLY

CALLS : PSHFPN, A9X
FIX 1
VECTOR

NOTES : THIS ROUTINE IS CALLED TWICE FOR 1 VECTOR, THAT IS ONCE FOR EACH AXIS (YAXIS KEEPS TRACK). IT ALSO DROPS VECTORS THAT GO BEYOND SCREEN EDGES.

DSPAGE/DSHOME

FUNCTION: DSPAGE = PAGES/HOMES CRT
DSHOME = HOMES CRT

OUTPUT: CLEARS FULL

USES: TMP1
TMPHY,TMPHX
TMPLY,TMPLX

CALLS: MOVE (ROUTINE INTERNAL TO CRTDRV)
DELY1S

DSPCPY/DSPCP2

FUNCTION: ISSUES A DISPLAY COPY COMMAND TO THE HARD COPY CONNECTOR.

CALLS: DRBUSY
WAIT.

NOTES: DSPCPY CHECKS TO SEE IF THE SCREEN IS ENABLED, IF IT IS NOT IT WILL SET THE COPY PENDING BIT.

DSPCHR

FUNCTION: SENDS ASCII CHARACTERS TO THE SCREEN AS IN CTLCHR EXCEPT IT CONVERTS ALL CONTROL CHARACTERS EXCEPT <CR> TO THE READABLE BACKSPACE UNDERLINE FORM.

INPUT: ACC. A = ASCII CHARACTER

CALLS: CTLCHR

DSPGIN

FUNCTION: PERFORM INPUT FUNCTIONS FROM THE DISPLAY AS DETERMINED BY THE SECONDARY ADDRESS.

INPUTS: A.SEC : CURRENT SEC. ADDR.
24. = GIN <= RETURN CURRENT POSITION IN GDU"S
13. = INPUT <= RETURN SCREEN SIZE IN GDU"S.

YAXIS : AXIS FLIP-FLOP
0 = RETURN X
<>0 = RETURN Y
NOTE: TOGGLE HANDLED BY DSPGIN ROUTINE.

POINT : POINTER TO RESULTING VALUE.

OUTPUTS: VALUE STORED AT POINT.

USES: YAXIS
TMPHX, TMPLX : READ TO GET CURRENT POSITION
TMPHY, TMPLY

CALLS: NIODEV
FLOAT 1
DOFP
PULFPN

NOTE: ONLY I/O VARIABLE USED IS A.SEC.

DSPOUT

FUNCTION: PERFORMS THE BASIC DISPLAY FUNCTIONS AS DICTATED BY THE CURRENT SECONDARY ADDRESS AND A BUFFER FULL OF INFORMATION.

INPUT: A.SEC : DEFINES WHAT IS TO BE DONE TO THE CHARACTERS IN THE BUFFER.

A.PTR : POINTER TO THE FIRST VALID CHARACTER IN THE BUFFER.

A.END : POINTER TO THE LAST VALID CHARACTER.

USES: A.PTR : MOVING POINTER.

CALLS: PCHAR
CTLCHR
DSPCPY
DOIT

ENABLE

FUNCTION: THIS ROUTINE IS USED TO CONTROL SYSTEM INTERRUPTIONS. THE ROUTINE WILL DECREMENT THE SYSTEM INTERRUPT COUNT AND IF IT BECOMES ZERO, SYSTEM INTERRUPTS CAN BE PROCESSED. THE SYSTEM MASK IS NOT ZERO IF OPERATIONS ARE IN PROGRESS THAT CAN NOT BE INTERRUPTED SAFELY. THESE OPERATIONS HAVE SYSTEM POINTERS OR CONTROL TABLES IN A STATE THAT IS NOT CONTROLLED.

RESTRICTIONS, NOTES: THIS ROUTINE CAN CALL ANY INTERRUPT PROCESSOR SO THE VARIABLE LIST AND ROUTINE LIST CAN NOT BE GENERATED. SEE ALSO, DISABLE AND SAFE.

ETOX/ZZETOX

FUNCTION: EXPONENTIAL EXP(X)

INPUT: X ON STACK

OUTPUT: EXP(X) ON STACK

IF EXP(X) OVERFLOWS OR UNDERFLOWS IT IS REPLACED BY THE LARGEST POSITIVE NUMBER OR ZERO, RESPECTIVELY, AND FREXP IS PLACED IN ERRCD.

USES:
R3
R1
R1+1
R3+1
TABPNT

CALLS:
PSHRET
PSHFPN
FPMUL
ZANS
MAXANS
SLN
NORVR
DOFP
FPNEG
FPDIV
RTRN

METHOD: MULTIPLIES X BY LOG 2 (E) THEN PARTITIONS

/X/ = N + (M+Y) / 16)

WHERE N AND M ARE INTEGERS. 2^N IS EASY, 2^(M/16) IS BY TABLE LOOKUP IN TABAD, AND 2^(Y/16) IS COMPLETED BY A RATIONAL APPROXIMATION OF THE FORM.

2^X = (Q(X^2) + XP(X^2)) / (Q(X^2) - XP(X^2))

FOR 0<= X <= 1/16, NO SOURCE IS GIVEN FOR THE APPROXIMATION; P AND Q ARE BOTH OF DEGREE 1.

FIX1

FUNCTION: FIX FLOATING POINT TO 16 BIT INTEGER.

IF INPUT IS NEGATIVE, FIX (ABS(A)) IS COMPUTED AND ERFIXN IS SET IN ERRCD. IF THE INPUT EXCEEDS 2/16-1, THE LARGEST POSITIVE INTEGER IS RETURNED AND ERFIXOV IS SET IN ERRCD. IF THE INPUT IS LESS THAN 0.5, ZERO IS RETURNED. OTHERWISE THE INPUT IS ROUNDED TO THE NEAREST UNSIGNED 16 BIT INTEGER.

INPUT: A FLOATING POINT NUMBER. ON ENTRY, IX POINTS TO THE TAG BYTE FOR THAT FP NUMBER.

OUTPUT: THE 16 BIT INTEGER IS PLACED IN THE HIGH ORDER BYTES OF THE INPUT MANTISSA, IX+3 AND IX+4.

FLOAT1 ALIAS FLOAT2

FUNCTION: FLOAT A 16 BIT INTEGER.

AN UNNUMERALIZED FP NUMBER WITH EXPONENT 48 IS FORMED, THEN NORM IS CALLED.

INPUT: TAGGED 2 BYTE INTEGER ON STACK

OUTPUT: TAGGED FP RESULT IN STACK

CALLS: PSHRET, NORM

FMTCLN

FUNCTION: TO OUTPUT ANY REMAINING DATA THAT IS SPECIFIED IN THE IMAGE STRING AFTER ALL LIST OUPUT IS COMPLETE.

INPUTS: ISP, ISL

OUTPUTS: SEE FUNCTION

USES: R0-R3

CALLS: UPCASE, PUTBYT

NOTES: THE CALLER OF FMTCLN MUST INSURE THAT THE INPUTS REMAIN VALID DURING THE DATA OUTPUT SEQUENCE.

FMTINI

FUNCTION: TO INITIALIZE THE FORMAT OUTPUT PROCESS BY CLEARING ALL FLAGS, SET CURSER = TO FIRST CHARACTER.

INPUTS: ISL

OUTPUTS: SEE FUNCTION

USES: <-

CALLS: <-

NOTES: <-

FMTPNT

FUNCTION: TO OUTPUT NUMERIC AND STRING DATA TYPES ACCORDING TO THE IMAGE STRING AND ALSO TO OUTPUT DATA AS SPECIFIED WITHIN THE IMAGE STRING ITSELF.

INPUTS: ISP, ISL, DP, DL, DT

OUTPUTS: OUTPUTS FORMATTED DATA THROUGH PUTBYT IN ACCORDANCE WITH ABOVE INPUTS.

USES: R0 - R3

CALLS: UPCASE, PUTBYT

NOTES: THE CALLER OF FMTPNT MUST INSURE THAT THE INPUTS STAY VALID DURING THE DATA OUTPUT SEQUENCE.

FPADD

FUNCTION: FP ADD

TWO FLOATING POINT OPERANDS A AND B ARE ON THE STACK ALONG WITH THEIR TAGS, B IS ON TOP, A IS NEXT. THE RESULT R:=A+B REPLACES THEM ON THE STACK. NO GUARD BITS ARE USED, UNDERFLOWS AND OVERFLOWS ARE HANDLED IN THE STANDARD NORM ROUTINE.

INPUT: 2FP OPERANDS IN STACK

OUTPUT: ONE FP RESULT IN STACK

USES: X0 TO SAVE SIGN (B)

CALLS: PSHRET, RTRN
A9X
SHMR (RIGHT SHIFT MANTISSA)
FRTST (PART OF NORM)
NORM

FPCMP

FUNCTION: COMPARE TWO FP NUMBERS ON STACK.

INPUT: X, ON TOP OF STACK.
Y, NEXT ON STACK.

OUTPUT: IN ACCA:
-1 IF X > Y
+1 IF Y > X
0 IF X = Y

USES: TABPNT

CALLS: PSHRET
A9X-1
CMPFPN
RTRN

FPDIV

FUNCTION: FP DIVIDE

COMPUTES FP QUOTIENT OF 2 FP NUMBERS IN THE STACK. THE RESULT R:=A/B REPLACES A AND B. OVERFLOW, UNDERFLOW AND DIVIDE BY ZERO ARE HANDLED IN NORM. NO GUARD DIGITS USED.

INPUT: B, ON TOP OF STACK, WITH TAG
A, NEXT IN STACK, WITH TAG

OUTPUT: R, ON TOP OF STACK

USES: T1, T2, X0,..X5, Y0,...Y5

CALLS: PSHRET
A9X
MAXANS (IN NORM)
FPRET
NORM

FPMUL

FUNCTION: FP MULTIPLY

FORMS FP PRODUCT OF TWO FP NUMBERS ON THE STACK. THE RESULT R:=A*B REPLACES A AND B. NO GUARD DIGITS ARE USED. UNDERFLOWS AND OVERFLOWS ARE HANDLED BY NORM.

INPUT: B, ON TOP OF STACK, WITH TAG
A, NEXT ON STACK, WITH TAG

OUTPUT: R, RESULT ON TOP OF STACK

USES: T1, T2, X0...X5, Y0,...,Y5

CALLS: PSHRET
A9X
ZANS (IN NORM)
NORM

FPNASC

FUNCTION: REDUCE A FLOATING PRINT NUMBER X TO A FRACTION F AND AN EXPONENT E SUCH THAT X = F . 10^E AND 1/10<=F<1.

INPUTS: THE FP NUMBER IS ON THE STACK. THE TABLES OF POWERS OF 10,
PLTAB AND NESTAB, ARE USED.

OUTPUTS: THE FP FRACTION MANTISSA, NOT NUMERALIZED IF 1/10<=F <1/2, IS IN X5...X5+5(X0).

THE EXPONENT E IS IN DEXP AND DEXP+1. IF X<0 THEN N5:="-"ELSE N5:=" ".

USES: X0...X5,Y1,Y0
N12, T4
TABPNT, MULPNT

CALLS: PSHRET
A9X
RTRN
CMPFPN
FPMUL
A8X
PULFPN
SHMR

FPNEG

FUNCTION: NEGATE A FLOATING POINT NUMBER: G = -F

INPUT: A FLOATING POINT NUMBER F ON TOP OF THE STACK.

OUTPUT: A FLOATING POINT NUMBER G = -F REPLACES F ON THE STACK.

CALLS: NONE

FPSUB

FUNCTION: FP SUBTRACT

REVERSES SIGN OF FP OPERAND ON TOP OF STACK AND GOES TO FPADD.

CALLS: FPADD

FUZZIE

FUNCTION: FUZZY COMPARE.

INPUTS: FP NUMBER B ON TOP OF STACK.
FP NUMBER A NEXT ON STACK.
FUZB, A GLOBAL FP NUMBER WITH THE ZERO-COMPARE FUZZY EXPONENT.

FUZA, A GLOBAL BYTE WITH THE NON-ZERO COMPARE FUZZ VALUE.

OUTPUTS: A AND B ARE DESTROYED.
A FP ZERO IS PLACED ON THE STACK.
ACC A AND T4 SET TO:

0 IF B FUZZY EQUAL A
+1 IF A > B
-1 IF A < B

USES: T4, TARPNT, FUZP

CALLS: FPSUB
CMPFPN
ZANS (IN NORM)
PSHRET

NOTES: FUZZY EQUAL OF NON-ZERO A TO 0.0 IS DEFINED BY COMPARING A TO A NUMBER WHOSE EXPONENT IS FUZB. IF A <= THAT NUMBER, THEN FUZZY EQUAL IS TRUE.

FUZZY EQUAL BETWEEN NON-ZERO A AND B IS DEFINED BY COMPUTING A-B AND COUNTING THE NUMBER OF NORMALIZE SHIFTS REQUIRED. IF THAT NUMBER IS >= FUZA, THEN FUZZY EQUAL IS TRUE.

GENCUR/DOIT

FUNCTION: GENERATES A WRITE-THRU CURSOR.

INPUT: CURSOR : ASCII CHARACTER TO FLASH.

USES: DISCNT - COUNTER

BLINK - TOGGLED TO GET BLINK ACTION. WHILE BLINK = 0, JUST IDLING FOR A WHILE.

CALLS: PCHAR

GETCHR

FUNCTION: EXTRACT ONE CHARACTER FROM THE CURRENT I/O DEVICE BUFFER. ALL DEVICES ARE HANDLED ON A BUFFERED BASIS. IF THE BUFFER IS EMPTY THEN A NEW BUFFER WILL BE REQUESTED FROM THE I/O DEVICE. A BUFFER REQUEST IS SATISFIED WHEN THE BUFFER BECOMES FULL OR AN END OF LINE CHARACTER IS ENCOUNTERED.

INPUTS: A. STAT : BFRSTT - BIT 5 <32.> SET IF BUFFER VALID (CONTAINS DATA).

A.PTR : POINTER TO NEXT VALID CHARACTER

A.END : POINTER TO FIRST NON-CHARACTER.

CRSTAT : BUFFFR STATUS FLAGS AS SFT BY I/O HANDLERS.

OUTPUTS: CHAR : CHARACTER READ
ACC.A : CHARACTER RFAD.
A.PTR : UPDATED
A.STAT : UPDATED.
CRSTAT : UPDATED

IF READ A DELIMITER CHARACTER THE CRVLD BIT WILL BE TURNED ON IN CRSTAT AND A <CR> WILL BE RETURNED AS THE CHARACTER REQUESTED.

ERRCD : SET IF ERROR IN I/O HANDLERS.

CALLS: NEWBFR

NOTE: ALL I/O SYSTEM VARIABLES ARE ASSUMED TO BE CORRECT.

IECIN

FUNCTION: TO SEND A BUFFER OF INFORMATION ALONG THE GPIB. IT WILL ALSO SEND MTA AND MSA IF REQUIRED.

INPUTS: A.STAT : BUSACT = BIT 4 <16.>
0 = SEND MTA AND MSA THEN SET BUSACT.
1 = ASSUME BUS SET UP.

A.PRIM : PRIMARY ADDRESS (MTA)

A.SEC : SECONDARY ADDRESS (MSA)

A.PTR : POINTER TO FIRST AVAILABLE CHARACTER POSITION IN BUFFER

A.MAX : POINTER TO LAST AVAILABLE CHARACTER POSITION IN BUFFER.

IOFUNC : CODE FOR PRESENT I/O FUNCTION, IF = READ <14.> THEN NO SEARCH FOR DELIMITERS WILL BE PERFORMED.

ETXCHR : 8 BIT END-OF-TEXT (OR EOF) CHARACTER. IF THIS CHARACTER IS DETECTED THE ETX BIT IN CRSTAT WILL BE SET AND INPUT FROM THE BUS WILL BE TERMINATED.

EOLCHR : 8 BIT END-OF-LINE CHARACTER. IF THIS CHARACTER IS DETECTED THE CRNORM BIT IN CRSTAT WILL BE SET AND INPUT FROM THE BUS WILL BE TERMINTED, AT LEAST UNTIL THIS ROUTINE IS CALLED AGAIN.

NULCHR : 7 BIT NULL CHARACTER. IF NULCHR >= 128. THEN NO CHARACTER WILL BE NULLED, OTHERWISE IF THE NULCHR IS DETECTED THE CHARACTER WILL NOT BE ENTERED INTO THE BUFFER.

CRSTAT : BUFFER STATUS FLAGS MUST BE 0 BEFORE CALLING THIS ROUTINE.

OUTPUTS: CRSTAT : BUFFER TERMINATION STATUS FLAG. POSSIBLE VALUES ARE ENUMERATED:

0 = BUFFER IS FULL AND NO TERMINATION CHARACTER WAS RECEIVED.

CREOI = AN EOI WAS RECEIVED
CRNORM = AN EOLCHR WAS RECEIVED
CRETX = AN ETXCHR WAS RECEIVED.

A.END : POINTER TO 1 PAST LAST VALID DATA CHARACTER.

NOTE: EOLCHR AND ETXCHR APE NOT VALID DATA CHARACTERS

IECIN (continued)

THEY ARE DELIMITERS. IN OTHER WORDS:

IF CRSTAT = 0 OR CREOI
A,END WILL POINT TO 1 PAST THE LAST VALID DATA CHARACTER.

IF CRSTAT = CRNORM OR CRETX
A,END WILL POINT TO THAT DELIMITER.

ERRCD : SET IF NO BODY LISTENING DURING ADDRESSING SEQUENCES.

USES: CHAR : TEMPORARY CHARACTER STORAGE.

CALLS:
ATNON
ATNOFF
INTSRC
IECSND
INTACP
IECRD
SCRM$

NOTES: THIS ROUTINE WILL SET UP THE PIA"S FOR BUS TRANSFERS BUT OTHER ROUTINES MUST BE USED TO GET PIA"S OFF OF THE BUS. UNADRS WILL NORMALLY PERFORM THIS FUNCTION.

IECOUT

FUNCTION: TO OUTPUT A BUFFER FULL OF DATA ON THE GPIB, ALSO SEND PRIMARY LISTEN ADDRESS AND SECONDARY ADDRESS IF NECESSARY.

INPUTS: A,STAT : BUSACT - BIT 4 <16.>
0 - SEND MLA, MSA AND SET BIT 4
1 - ASSUME ALREADY ADDRESSED

A,PRIM : PRIMARY ADDRESS (MLA)

A,SEC : SECONDARY ADDRESS (MSA) NOTE: IF = 32 DON"T SEND A SEC. ADDRESS.

A,PTR : POINTING TO FIRST DATA CHARACTER

A,END : POINTING TO LAST DATA CHARACTER

NOOUT : SPDIT - BIT 7 <128>
0 - SEND DATA
1 - SEND DATA BUT ALSO SEND EOI WITH LAST BYTE.

OUTPUT: ERRCD SET IF NO DEVICES LISTENING TO BUS, NRFD AND NDAC SENSED HIGH.

USES: A,PTR AS A MOVING POINTER

CALLS: ATNON
INTSRC
IECSND
ATNOFF
EOION
EOIOFF

NOTE: THIS ROUTINE WILL INITIALIZE THE PIA"S TO GET ON THE BUS BUT SOME OTHER ROUTINE MUST GET OFF THE BUS. THE UNADRS ROUTINE WILL GENERALLY PERFORM THIS FUNCTION. THE ROUTINES DIRECTLY RESPONSIBLE FOR GETTING OFF THE BUS ARE

IECOFF AND IECIFC.

INAGTE

FUNCTION: CONVERTS AN INTEGER TO AN ASCII STRING WHERE THE STRING IS STORED IN R15-R20 AND HAS A NULL STORED AFTER THE LAST CHARACTER.

INPUTS: INDEX REG. IS INTEGER

OUTPUTS: STRING STORED IN R15 - R20 WITH A NULL AFTER THE LAST CHARACTER, R1 WILL POINT TO FIRST CHARACTER.

USES: R0, R1, R2, R15-R20

CALLS: INAITT

NOTE: THE ACTUAL STRING WILL START AT R16 BUT PART OF R15 IS USED BY INAITT.

INAITT/FPAITT

FUNCTION: INAITT CONVERT AN INTEGER TO AN ASCII STRING, FPAITT CONVERT FPN IN FPC TO AN ASCII STRING.

INPUTS: R0 - 2 BYTES : INTEGER TO CONVERT

R1 : POINTER TO FIRST CHARACTER OF RESULT STRING.
NOTE: CHARACTER R1-1, WILL BE CLEARED.

R2 : MAX. LENGTH OF STRING <72.

FPC : IF CALLING FPAITT THIS CONTAINS THE FPN.

OUTPUTS: R2 : POINTER TO LAST CHARACTER OF RESULT STRING.
R1 : POINTER TO FIRST CHARACTER OF RESULT STRING.

USES: FPC, NOUT
IOFLGS

CALLS: FLOAT1
PRIVAL
PULFPN
PSHFPN

INITMT

FUNCTION: INITIALIZE MAGTAPE PIA'S, REWIND IF NECESSARY, DISABLE CRT, DISABLE ABORT.

OUTPUTS: ERRCD : SET IF NO CART. PRESENT.
MTSTAT <- MTSTT2 <- RESET IF REWIND NEW TAPE
DSPSTT <- 128. ,, DISABLES DISPLAY.

CALLS: DRBUSY
DISBLE
PUPTAP
REWIND

INPSTG

FUNCTION: INPUT <INPUT> A STRING VARIABLE FROM THE CURRENT I/O DEVICE

INPUTS: POINT : POINTER TO FIRST CHARACTER POSITION TO RECEIVE DATA.

LENGTH : DIMENSION MAXIMUM LENGTH OF STRING.

OUTPUTS: STRING STORED STARTING AT POINT
CHRCNT : ACTUAL LENGTH READ.
ERRCD : SET IF ERRORS IN HANDLERS.

USES: COLCNT
TCOL - 2 BYTES
CRSTAT

CALLS: GETCHR

NOTES: ALL NORMAL I/O SYSTEM VARIABLES MUST BE SET UP CORRECTLY.

INPVAL

FUNCTION: INPUT <INPUT> A VALUE FROM THE CURRENT I/O DEVICE.

INPUTS: A,PRIM : CHECKS IF CURRENT DEVICE IS THE DISPLAY, WHICH REQUIRES SPECIAL HANDLING.

POINT : POINTER TO RESULT VALUE LOCATION.

OUTPUTS: VALUE STORED IN RESULT LOCATION.
ERRCD : SET IF ERROR IN HANDLERS.

USES: CRSTAT
A,PTR
DIGFLG
FFLAG

CALLS: DSPGIN
A9X
GETCHP
ASCFPN
PULFPN

NOTES: ALL NORMAL I/O SYSTEM VARIABLES MUST BE SET CORRECTLY.

INT

FUNCTION: GETS THE INTEGRAL PART OF A FLOATING POINT NUMBER, RETURNING A FLOATING POINT NUMBER. DOES AN ALGOL-FORTRAN STYLE FIX:

INT(1.5)=1,INT(-1.5)=-2

INPUT: FP NUMBER IN STACK

OUTPUT: FP NUMBER IN STACK

USES: R1

CALLS: PSHRET
ZANS
A9X
PSHFPN
FPNEG
RTRN

INTMLA

FUNCTION: INTMLA MULTIPLIES TWO 16 BIT UNSIGNED INTEGERS AND PRODUCES A 32 BIT UNSIGNED RESULT AND THEN ADDS TO THE RESULT A 16 BIT UNSIGNED INTEGER.

INPUTS: ONE INTEGER IS IN R6.
THE SECOND INTEGER IS IN THE "X" REGISTER.
THE INTEGER TO BE ADDED TO THE RESULT IS IN THE "A" AND "B" REGISTERS WITH "A" BEING THE HIGH ORDER BYTE.

OUTPUTS: THE OUTPUT IS IN R6 AND R7 WITH R6 BEING THE HIGH ORDER BYTE AND R7+1 THE LOW ORDER BYTE (R6, R7 =(R6*X)+AB).

THE "X" REGISTER IS LOADED WITH R6 JUST BEFORE EXIT SO THE HIGH ORDER 16 BITS MAY BE CHECKED ON RETURN BY A BEQ OR BNE INSTRUCTION.

USES: R6, R7

CALLS: <-

NOTES: NO ERRORS OR OVERFLOW POSSIBLE.

INTMLC

FUNCTION: INTMLC MULTIPLIES TWO 16 BIT UNSIGNED INTEGERS AND PRODUCES A 32 BIT UNSIGNED RESULT.

INPUTS: ONE INTEGER IS IN R6.
THE SECOND INTEGER IS IN THE "X" REGISTER.

OUTPUTS: THE OUTPUT IS IN R6 AND R7 WITH R6 BEING THE HIGH ORDER BYTE AND R7+1 THE LOW ORDER BYTE (R6, R7=R6*X)
THE "X" REGISTER IS LOADED WITH R6 JUST BEFORE EXIT SO THE HIGH ORDER 16 BITS MAY BE CHECKED ON RETURN BY A BEQ OR BNE INSTRUCTION.

USES: R6, R7

CALLS: <-

NOTES: NO ERRORS OR OVERFLOW POSSIBLE.

IOCLNR

FUNCTION: CLEANS UP THE STACK AFTER THE PROCESSING OF I/O LISTS BY THE I/O SYSTEM. THE I/O SYSTEM PUSHES A SECOND EOLTG BEFORE IT BUILDS A NORMAL LIST ON TOP OF THE POST FIX INFORMATION. THIS ROUTINE SEARCHES FOR THE FIRST EOLTG AND RETRIEVES THAT RETURN ADDRESS. IT THEN SEARCHES FOR A SECOND EOLTG (THE ONE THE EVALUATOR PLACED) AND MOVES THE STACK POINTER TO THAT EOLTG. THE ROUTINE RETURNS TO THE RETURN ADDRESS RETRIEVED EARLY WITH THE FIRST EOLTG. NOTE: THIS ROUTINE IS JSR'D TOO, BUT THE RETURN ADDRESS IS NEVER USED EXCEPT FOR MAYBE DEBUGGING.

USES: DREXTA, R0

CALLS: LOCTG
DREXTA

KEYIN

FUNCTION: SET THE KEYBOARD UP FOR RESPONSE TO THE INPUT STATEMENT, COMPLETE WITH FLASHING\.

INPUT: A,SEC ; USED TO DETERMINE IF INPUT REQUESTED.

OUTPUT: A FULL DATA BUFFER DETIMITED BY A <CR>.

USES: CURSOR
BFLAG

CALLS: NIODEV
CIDLE

NOTE: A, END WILL BE SET UP BY TYPIN WITH THE ASSUMPTION THAT THE EDIT BUFFER (EDTBFR) IS BEING USED.

LOCTG (LOCATE TAG)

FUNCTION: THIS ROUTINE IS USED TO LOCATE A GIVEN TAG ON THE STACK. THE STACK ENTRIES ARE SEARCH FOR THE TAG AND IF IT IS NOT FOUND IN THE STACK THE RESULT POINTER IS SET TO ZERO. A PSUEDO STACK POINTER (R0) IS USED FOR THE STARTING POINT OF THE SEARCH.

INPUTS: R0 IS A POINTER TO THE STARTING LOCATION ON THE STACK-1.
ACC-A IS THE TAG YOU WISH TO FIND.

OUTPUTS: R0 IS UPDATED TO POINT TO THE FOUND ENTRY-1.

USES: R0 AND R1

CALLS: BACKUP.

LOCTGR

FUNCTION: LOCATE A TAG IN A RANGE OF VALUES. THIS ROUTINE IS VERY SIMILAR TO LOCTG EXCEPT IT WILL FIND THE FIRST OF ANY TAGS IN A GIVEN RANGE OF TAG VALUES.

INPUTS: R0 IS THE PSEUDO STACK POINTER, THE ADDRESS OF TAG-1.

ACC-A IS THE LOW VALUE.
ACC-B IS THE HIGH VALUE.

OUTPUTS: R0 HAS THE ADDRESS OF THE FIRST FOUND TAG-1.

USES: R0 AND R1

CALLS: BACKUP.

LOG/LGN/ZZLOG

FUNCTION: COMPUTE LOGE(X) OR LOG10(X)

INPUTS:
1. X ON STACK
2. GLOBAL CTKN. IF CTKN=LGNCOD, LOG10 IS COMPUTED; OTHERWISE LOGE.

OUTPUTS: LOG(X) ON STACK. IF X<=0, X IS RETURNED AND ERLOG IS PUT IN ERRCD.

USES: R3, R3+1

CALLS:
PSHPET
RTRN
DOFP
FLOAT 2 (=FLOAT 1)
FPMUL
FPADD
PSHFPN

METHOD: COMPUTES LOG 2(X) THEN MULTIPLIES BY LOGE(2) AND OPTIONALLY LOG10(E). THE EXPONENT OF X IS CONVERTED TO AN INTEGER; THE FRACTION IS CHANGED TO THE RANGE SQRT(0.5)<=W<=SQRT(2) AND T=(W-1)/(W+1) IS COMPUTED.

THEN LOG 2(W)=T*P(T'2)

WHERE P IS A POLYNOMIAL OF DEGREE 7 IN T'2, NO SOURCE IS GIVEN FOR P WHICH IS ASSERTED TO BE CHEBYSHEV.

MIN,MAX

FUNCTION: FIND EITHER MIN OR MAX OF TOP TWO FP NUMBERS ON STACK.

INPUT:
X, A FP NUMBER ON TOP OF STACK.
Y, A FP NUMBER NEXT ON STACK.
BOTH INPUTS ARE POPPED.

OUTPUT: MIN(X,Y) OR MAX (X,Y) ON STACK

CALLS:
PSHRET
FPCMP
A9X-1
PULFPN
A9X
RTRN

MKFILE

FUNCTION: MARKS MAGTAPE FILES ON THE INTERNAL MAGTAPE.

INPUTS: R8+1 : NUMBER OF FILES TO MARK

R9 : NUMBER OF RECORDS PER FILE - NOTE: MUST BE 3 OR GREATER.

A,STRT : POINTER TO FIRST CHARACTER LOCATION IN THE RECORD. NOTE: MUST BE MTBFR.

FILLOC : PRESENT FILE NUMBER

MTSTT2 : MTFFST - BIT 7 (128) HEAD IN GAP BIT.

MTSREG : TAPE CONFIGURATION STATUS. LENGTH/ CHECK SUM/ HEADER

OUTPUTS: ERRCD : SET IF TOO MANY FILES OR END OF MEDIUM.

FILLOC : MODIFIED TO NEW LOCATION <- NOTE: EXITS WITH HEAD JUST BEFORE #LAST# FILE.

USES: R0, R1, R2, R10
MTPTR
MTMAX
FILEND

CALLS: NFR8
MTNULL
MODT88
INAITT
MTCHKS
MARKS
REWIND
MTCLOS
PUPTAP

NOTE: MAGTAPE MUST BE INITIALIZED.
ALSO, BEFORE CALLING, A TEST FOR THE WRITE LOCKED CONDITION SHOULD BE MADE. THE TAPE IS WRITE LOCKED IF PIAMTA BIT 3 (8.) IS SET. IF A FILE IS OPEN MTSTAT - MTFPOS: BIT 6(64) IS SET THEN THIS IS ALSO AN ERROR AND THE ROUTINE SHOULD NOT BE CALLED.

MODMTH

FUNCTION: UPDATES OR MODIFIES THE MAGTAPE HEADER RECORD IN MEMORY.

INPUTS: A,STRT : POINTER TO BEGINNING OF HEADER RECORD.

ACC,B : INDEX OFF OF A,STRT TO POSITON TO MODIFY.

INDEX REG. : POINTER TO NEW HEADER TEXT.

ACC,A : LENGTH OF NEW HEADER TEXT.

OUTPUTS: MODIFIED HEADER RECORD IN MEMORY.

USES: R0,R1

MTAFIN

FUNCTION: FINDS AND OPENS A FILE ON THE INTERNAL MAGTAPE.

INPUTS:
ACC,B ; FILE TO BE LOCATED
FILLOC-2 BYTES ; PRESENT FILE POSITION.
MTSREG ; TAPE CONFIGURATION FLAGS.
MTSTAT ; TAPE STATUS
MTSTT2 ; TAPE STATUS
A,STRT ; POINTER TO FIRST CHARACTER IN TAPE BUFFER.
A,MAX ; POINTER TO LAST CHARACTER IN TAPE BUFFER.
NOTE: IF 128. BYTE RECORDS A,STRT+128. = A,MAX

OUTPUTS:
FILLOC ; SET TO NEW FILE POSITION
MTSTAT ; SET TO NEW STATUS
MTSTT2 ; SET TO NEW STATUS
ERRCD ; SET IF ERROR
RECCNT-2 BYTES ; OPEN FILE - AVAILABLE RECORD COUNT.

NOTE : IF NO HEADER MODE RECCNT = 65535. MTMAX = A,MAX, MTPTR = A,STRT

USES:
TABPTR
FILEND
R0, R1, FPC

CALLS:
MTCLOS
REWIND
SERCHS
MTREAD
AFPITT
FIX1
A14X
MTBINT

NOTE: MAGTAPE MUST BE INITIALIZED.

MTCLOS

FUNCTION: CLOSES OPEN MAGTAPE FILES; WILL WRITE LAST RECORD IF THE MAGTAPE BUFFER CONTAINS WRITTEN INFORMATION.

INPUTS: MTSTT2; MAGTAPE STATUS - INCLUDES WRITE BIT.

MTPTR; POINTING TO NEXT CHARACTER. SPACE IN MAGTAPE BUFFER, ONLY NEEDED IF WRITING.

OUTPUTS: MTSTT2 ; MAGTAPE STATUS
MTSTAT<-0 ; MAGTAPE STATUS

CALLS: MTBSWP
MTWRIT
A9X

NOTE: MAGTAPE MUST BE INITIALIZED.

MTPADR

FUNCTION: INITIALIZES MAGTAPE AND CHECKS FOR VALIDITY OF I/O OPERATION VERSUS PRESENT MAGTAPE STATUS AND FILE. FILE HEADERS WILL BE MODIFIED IF APPROPRIATE.

INPUTS: A.SEC ; SECONDARY ADDRESS FOR PRESENT I/O OPERATION, THIS WILL DETERMINE WHAT ACTION WILL BE TAKEN.

OUTPUTS: ERRCD ; SET IF ERROR.
MAGTAPE INITIALIZED FOR MAGTAPE OPERATORS, CRT DISABLE, ABORT DISABLED.

CALLS: INITMT
BFRALC
XFRSCN
MTPPRI, MTPINP
MTPOLD, MTPSAV
MTPRD, MTPWRI

MTPIN

FUNCTION: GRABS CHARACTERS FROM MAGTAPE BUFFER FOR BASIC I/O.

INPUTS: MTPTR ; POINTING TO NEXT VALID CHARACTER.

MTMAX ; POINTING TO LAST VALID CHARACTER.

A.PTR ; POINTING TO FIRST CHARACTER OF RESULT STORAGE.

A.MAX ; POINTING TO LAST CHARACTER OF RESULT STORAGE.

MTSTT2 ; MAGTAPE STATUS

RECCNT ; AVAILABLE RECORD COUNT.

IOFUNC ; CURRENT I/O FUNCTION CODE IF = 14. THEN ASSUME DOING A READ AND WILL FILL FROM A.PTR THRU A.MAX. OTHERWISE SEARCH FOR LINE DELIMITERS AND SET UP A.END ACCORDINGLY.

ETXCHR ; INPUT EOF CHARACTER

NULCHR ; INPUT NULL CHARACTER

EOLCHR ; INPUT END-OF-LINE CHARACTER.

OUTPUTS: A.END ; POINTING TO DELIMITER CHARACTER OR IF ONE WAS NOT FOUND BEFORE REACHING A.MAX THEN IT WILL BE POINTING TO 1 PAST THE LAST VALID CHARACTER.

CRSTAT ; 0 IF NO DELIMITER FOUND
; CRNORM IF EOLCHR FOUND
; CREOT IF ETXCHR FOUND
NOTE: A <CR> WILL BE PLACED OVER THE ETXCHR

PNDEOF ; SET TO 128. IF LOGICAL OR PHYSICAL EOF ENCOUNTERED.

MTPTR ; POINTING TO NEXT VALID CHARACTER IN MAGTAPE BUFFER.

ERRCD; SET IF ERROR.

CALLS: MTRBFR
MTBSWP
MTREAD
MTRINT
PULFPN
SCRMS

NOTE: MAGTAPE MUST BE INITIALIZED.

MTPOUT

FUNCTION: TO TRANSFER DATA FROM BASIC"S I/O BUFFER TO THE MAGTAPE BUFFER AND EVENTUALLY TO THE MAGTAPE.

INPUTS: MTSTT2 ; MAGTAPE STATUS.
RECCNT ; AVAILABLE RECORD COUNT
A,PTR ; POINTING TO FIRST CHARACTER TO XFER.
A,END ; POINTING TO LAST CHARACTER TO XFER.
MTPTR ; POINTING TO NEXT AVAILABLE SLOT IN MAGTAPE BUFFER.
MTMAX ; POINTER TO LAST AVAILABLE SLOT IN MAGTAPE BUFFER.

OUTPUTS: MTSTT2 ; NEW MAGTAPE STATUS
PNDFLG =128,; IF RECCNT = 0 WHEN CALLED.

CALLS: BAKARS
MTWRIT
PULFPN

NOTE: MAGTAPE MUST BE INITIALIZED.

NOTE: THIS ROUTINE WILL BACK UP TO LAST RECORD IF THE RECORD IN MEMORY WAS PARTIALLY READ.

MTREAD

FUNCTION: READS A RECORD OFF OF THE MAGTAPE INTO THE MAGTAPE BUFFER.

INPUT: A.STRT ; POINTING TO FIRST CHARACTER OF MAGTAPE BUFFER.

A.MAX ; POINTING TO LAST CHARACTER OF MAGTAPE BUFFER. NOTE: IF USING 128. BYTE RECORDS A.MAX MUST BE SET ACCORDINGLY.

MTSTT2 ; MAGTAPE STATUS

OUTPUTS: RECORD STORED IN BUFFER

MTSTT2 ; NEW MAGTAPE STATUS

RECCNT <= 0 IF PHYSICAL END OF FILE ENCOUNTERED (8NFR"S)

FILLOC ; MODIFIED IF RECCNT <=0

MTERR ; NUMBER OF RE-READS IF READ ERRORS DETECTED (MAX 9).

ERRCD ; SET OF FATAL READ (10 RE-READS)

USES: MTPTR
MTMAX

CALLS: BAKARS
READRS
MTWAIT
PUPTAP
DELY1S
MTCHKS

NOTE: MAGTAPE MUST BE INITIALIZED.

MTWRIT

FUNCTION: WRITE A BUFFER ONTO THE MAGTAPE.

INPUTS: RECCNT ; AVAILABLE RECORD COUNT FOR OPEN FILE. NOTE: IF IT IS ALREADY 0 DON"T CALL THIS ROUTINE.

A,STRT ; POINTING TO FIRST CHARACTER

A,MAX ; POINTING TO LAST CHARACTER; MUST BE ON A RECORD BOUND.

MTSREG ; TAPE CONFIGURATION.

OUTPUTS: RECCNT ; MODIFIED
MTSTT2 ; MAGTAPE STATUS
EPRCD ; SET IF WRITE LOCKED.

USES: MTPTR, MTMAX, MTFLGS

CALLS: MTCHKS
WPRS
MTWAIT
WREND.
PUPTAP

NOTE: MAGTAPE MUST BE INITIALIZED

NEWBFR

FUNCTION: GET A NEW BUFFER FULL OF DATA FOR INPUT PROCESSING.

INPUTS: PPMODE : PERCENT MODE FLAG - SET IF IN PERCENT MODE

A,STRT : POINTER TO FIRST CHARACTER OF THE BUFFER.

A,MAX : POINTER TO THE LAST CHARACTER OF THE BUFFER. NOTE: ONE POSITION PAST THAT INDICATED BY A,MAX MAY BE USED FOR DELIMITER STORAGE. SO WATCH OUT.

OUTPUTS: CRSTAT : SET BY I/O HANDLERS

A,PTR : SET TO FIRST CHARACTER

A,END : SET TO LAST VALID CHARACTER. (NOTE: END OF LINE DELIMITERS ARE NOT CONSIDERED TO BE VALID CHARACTERS.

A,STAT: BFRSTT - BIT 5 <32.> SET
ERRCD: SET BY I/O HANDLERS

USES: SCRATCH AREA WHEN PURGING UNDER PERCENT MODE.

CALLS: NXBFR <- REALLY PART OF NEWBFR
PSHFPN
PULFPN
XFRCTL
IECIN
KEYIN
NIODEV
MTPIN

NOTE: ALL NORMAL I/O SYSTEM VARIABLES MUST BE SET CORRECTLY.

NFR8

FUNCTION: TO SEARCH IN REVERSE FOR AN 8NFR FILE MARK.

CALLS: BAKARS
TST8NF

NOTE: MAGTAPE MUST BE INITIALIZED.

NORM

FUNCTION: NORMALIZE FLOATING POINT NUMBER ON STACK, CHECKS FOR OVERFLOW, UNDERFLOW, AND ZERO RESULT. UNDERFLOW GOES TO ZERO. OVERFLOW IS REPLACED BY THE LARGEST POSSIBLE NUMBER.

NORM IS REACHED BY A JMP AFTER PUTTING THE RETURN ADDRESS IN FNRET; I.E., LAST INSTRUCTION IN NORM IS A JMP TO RTRN.

INPUT: FP NUMBER ON STACK.

OUTPUT: NORMALIZED OR ZERO FP NUMBER ON STACK. IN THE EVENT OF OVERFLOW, ERFPOV IS PLACED IN ERRCD. THE NUMBER OF LEFT SHIFTS REQUIRED TO NORMALIZE IS KEPT IN THE GLOBAL FUZR FOR SUBSEQUENT USE IN FUZZY COMPARE.

OUTBFR

FUNCTION: ATTACH DEVICE HANDLER TO OUTPUT CURRENT OUTPUT BUFFER, ALSO CONVERT CONTROL CHARACTERS TO #LIST# FORMAT IF REQUIRED.

INPUTS: NOOUT: SNDIT : BIT 7 (128.) SEND EOI WITH LAST BYTE IF SET.

. LSTFMT : BIT 7 (2.) SET IF #LIST# FORMAT CONTROL CHARACTER.

. NOWRIT : BIT 0 (1.) SET IF SHOULDN'T WRITE THE BUFFER. AND ERROR WILL BE SET.

A.PTR : POINTER TO FIRST CHARACTER SLOT.

A.END : POINTER TO MOST RECENTLY USED CHARACTER SLOT USED BY PUTBYT.

IOFUNC : CODE FOR PRESENT I/O FUNCTION.

A.STRT : POINTER TO FIRST CHARACTER SLOT.

IECRLF : CR VS. CR/LF FLAG.

OUTPUTS: A.PTR, A.END; INITIALIZED FOR NEW BUFFER.

ERRCD ; SET IF ERROR IN I/O HANDLER, OR NOWRIT BIT SET IN NOOUT.

USES: SCRTCH SPACE

CALLS: SCRMO
XFRCTL
FILEOT
NIODEV
IECOUT
DSPOUT
MTPOUT
NOACT

NOTE: ALL I/O SYSTEM VARIABLES MUST BE SET CORRECTLY.

PCHAR

FUNCTION: DISPLAY A CHARACTER ON THE CRT, THIS INCLUDES CONTROL CHARACTERS.

INPUTS: ACC, A - ASCII CHARACTER.

CALLS:
DRBUSY
WAIT.
PCHR.1-THE ACTUAL CHARACTER POINTER
DSPCPZ
PAGE - INTERNAL TO CRTDRV
HOME - INTERNAL TO CRTDRV
MTPRWD
CRTRST

NOTES: THE STATUS OF THE DISPLAY IS KEPT IN THE VARIABLE DSPSTT. DSPSTT CONTAINS THE FOLLOWING INFORMATION IN THE FORM OF A BIT PATTERN.

DSPDIS - SET IF DISPLAY DISABLED
HOMFLG - SET IF HOME PENDING
PAGFLG - SET IF PAGE PENDING
RWNDFG - SET IF REWIND PENDING
CPYFLG - SET IF COPY PENDING.

PCHR.1

FUNCTION: THIS ROUTINE PAINTS THE CHARACTERS ON THE CRT. THEY ARE STORED OR WRITTEN THRU AS DISCRIBED BY BLINK. ALSO DISCUSSES THE EXTERNAL ROM PACK CHARACTER POINT CAPABILITY.

INPUT: BLINK : 0 - STORE MODE
1 - WRITE-THRU MODE

NOTES: 1) TO USE THE EXTERNAL CHARACTER FONT CAPABILITY ONE MUST STORE THE BANK NUMBER AND SERVICE ROUTINE ADDRESS IN THE 3 BYTE CHRVCT VARIABLE. CONTROL WILL BE PASSED TO THIS ROUTINE ANYTIME A PRINTABLE CHARACTER REQUEST IS PROCESSD THRU PCHR.1. PLEASE REFER TO THE CODE IN CRTDRV FOR MORE INFORMATION.

2)DON'T BLINK CONTROL CHARACTERS.

PRINTF

THE MODULE PRINT F HAS SEVERAL ROUTINES AND THE MODULE"S FUNCTION WITHIN THE SYSTEM IS TO HANDLE CONVERSIONS FROM INTERNAL FORMAT TO ASCII FOR BOTH NUMBERS AND STRINGS.

FOR NUMBERS, A TRUE CONVERSION FROM BINARY FLOATING POINT TO ASCII IS MADE. FOR STRINGS, IT USUALLY ENTAILS TAKING THEM DIRECTLY FROM MEMORY AND OUTPUTTING THEM.

THERE ARE TWO DISTINCT PROCESSES INVOLVED:

1)THE FIRST IS WHEN CONVERSIONS ARE GOVERNED BY AN IMAGE STRING SUPPLIED BY THE USER.

2)THE SECOND IS WHEN CONVERSIONS ARE GOVERNED BY DEFAULT IMAGE STRING

ALL OUTPUT FOR BOTH PROCESSES IS PASSED THROUGH AN EXTERNAL ROUTINE - PUTBYT; THE USER OF PRINTF MUST HAVE AN INTIMATE KNOWLEDGE OF PUTBYT SINCE, BEFORE PRINTF CAN BE USED, PUTBYT MUST BE SET UP PROPERLY.

ALSO, FOR NUMBER OUTPUT, FPNASC MUST BE CALLED FOR EACH NUMBER OUTPUT BEFORE BEFORE PASSING CONTROL TO PRINTF EACH TIME.

PRINTF CONSISTS OF THE FOLLOWING USER CALLABLE ROUTINES:

A)FMTINI (FORMAT INITIALIZE)
B)FMTPNT (FORMAT PRINT; USING LIST OUTPUTS)
C)DEFPNT (DEFAULT PRINT; NON-USING OUTPUTS (PRINT LIST, STR, ETC))
D)FMTCLN (USED TO OUTPUT ANY REMAINING SPECIFIER"S THAT DO NOT REQUIRE DATA ITEMS FROM THE PRINT LIST)

THE DESCRIPTIONS FOR EACH ROUTINE FOLLOW THIS INTRODUCTORY VERBAGE.

THE GENERAL PROCEDURE FOR OUTPUTTING USING IMAGE STRING SUPPLIED CONTROL IS AS FOLLOWS:

1. CALL FMTINI TO INITIALIZE FMTPNT/FMTCLN.
2. SET UP PUTBYT.
3. FOR EACH STRING OUTPUT ENCOUNTERED, SIMPLY CALL FMTPNT.
4. FOR EACH NUMERIC OUTPUT ENCOUNTERED, CALL FPNASC FOLLOWED BY A CALL TO FMTPNT.

PRINTF (continued)

5. WHEN ALL ITEMS IN THE LIST ARE EXAUSTED, CALL FMTCLN.

THE GENERAL PROCEDURE FOR DEFAULT PRINTING IS (OUTPUT NOT CONTROLLED BY A USERS IMAGE STRING):

1. SET UP PUTBYT.
2. FOR EACH STRING OUTPUT ENCOUNTERED, CALL DEFPNT.
3. FOR EACH NUMERIC OUTPUT ENCOUNTERED, CALL FPNASC FOLLOWED BY A CALL TO DEFPNT.

THE NUMBER OF ERROR CONDITIONS ARE NUMEROUS AND ARE DEPENDANT ON THE IMAGE STRING AND DATA VALUE RELATIONSHIPS AS WELL AS SYNTAX OF THE IMAGE STRING. A DETAILED LIST OF ERRORS THAT CAN OCCUR CAN BE FOUND IN THE USERS MANUAL.

THE COMMON COMMUNICATON LOCATIONS ARE AS FOLLOWS:

ISP - IMAGE STRING POINTER; 2 BYTE IMAGE STRING POINTER; POINTS TO THE FIRST CHARACTER IN THE IMAGE STRING AT ALL TIMES.
ISL - IMAGE STRING LENGTH; 2 BYTE INTEGER EQUAL TO THE NUMBER OF CHARACTERS IN THE IMAGE STRING.
DT - DATA TYPE; SINGLE BYTE FLAG WHERE DT=0 FOR STRING AND 1 FOR NUMERIC;DT=1 IMPLIES FPNASC HAS BEEN CALLED.
DP - DATA POINTER; 2 BYTE STRING POINTER; POINTS TO THE FIRST CHARACTER IN THE STRING.
DL - DATA LENGTH; 2 BYTE INTEGER EQUAL TO THE NUMBER OF CHARACTERS IN THE STRING.

PRISTG

FUNCTION: PRINT A STRING. IF NORMAL XFER THE STRING WILL BE ENTERED INTO THE OUTPUT BUFFER.

INPUTS: POINT ; POINTER TO STRING
LENGTH ; LENGTH OF STRING

USES: DP, DL, DT

CALLS: PRISPC

NOTE: ALL I/O SYSTEM VARIABLES MUST BE SET UP CORRECTLY.

PRIVAL

FUNCTION: PRINT A VALUE. IF A NORMAL XFER THE VALUE WILL BE ENTERED AS ASCII CHARACTERS IN THE OUTPUT BUFFER.

INPUTS: POINT ; POINTS TO VALUE

USES: DT

CALLS: PRISPC

NOTE: ALL NORMAL I/O SYSTEM VAIABLES MUST BE SET UP CORRECTLY.

PSHFPN

FUNCTION: PUSH FLOATING

(1) THE FLOATING PRINT NUMBER IN IX+0,..,IX+7 IS PUSHED ON THE STACK.

(2) #VALTG# TAG IS PUSHED ON THE STACK.

(3) RETURN TO CALLER VIA DREXTB.

INPUTS: FLOATING POINT NUMBER POINTED TO BY IX, TOP 2 BYTES OF STACK.

OUTPUTS: FLOATING POINT NUMBER ON THE STACK. TAG VALTG ON STACK.

PSHRET

FUNCTION: THIS ROUTINE IS USED TO SAVE THE RETURN ADDRESS OFF THE SYSTEM STACK ONTO A SECONDARY TEN ELEMENT STACK.

INPUTS: ADDRESSES ON THE STACK.

OUTPUTS: NEW ENTRY ON SECONDARY STACK.

PULFPN

FUNCTION: POP FLOATING

(1) THE TAG IS THROWN AWAY.

(2) THE FLOATING POINT NUMBER ON THE STACK IS POPPED TO IX+0...IX+7.

(3) RETURN VIA DREXTB.

INPUT STACK: (TOP)

1 BYTE TAG
8 BYTES FP NUMBER

OUTPUT: FP NUMBER IN IX+0...IX+7

PUTBYT

FUNCTION: ENTERS A BYTE INTO THE CURRENT OUTPUT BUFFER. IF THAT BUFFER BECOMES FULL OR AN EOLCHR<CR> WAS IN THE BUFFER THEN THAT BUFFER IS TRANSMITTED TO THE CURRENT I/O DIVICE AND A FRESH BUFFER IS ESTABLISHED.

INPUTS: ACC.A ; THE CHARACTER TO BE PUT.
A.END ; POINTER TO LAST CHARACTER PUT.
A.MAX ; POINTER TO LAST SLOT IN THE OUTPUT BUFFER.
TABCNT ; CHARACTER COUNTER FOR PRINTF

OUTPUTS: ERRCD ; SET IF ERROR IN I/O HANDLERS
TABCNT ; MODIFIED CHARACTER COUNTER.
A.END ; UPDATED.

CALLS: OUTBFR

NOTE: ALL I/O SYSTEM VARIABLES MUST BE SET UP CORRECTLY.

QCROSS

FUNCTION: QCROSS IS INVOLKED AS A RESULT OF THE #POINTER# COMMAND. ITS FUNCTION IS TO POSITION THE GRAPHIC POINTER IN RESPONSE TO THE JOY STICK INPUT AND EXIT THE PROCESS UPON LEGAL INPUT FROM THE KEYBOARD.

INPUTS: ANALOG VOLTAGE FROM JOY STICK AND TERMINATING CHARACTER FROM KEYBOARD.

OUTPUTS: QCROSS CONTINUOUSLY SLEWS THE CRT DA"S SO AS TO TRACK THE JOYSTICKS ANALOG INPUT VOLTAGE UNTIL SUCH TIME AS A LEGAL CHARACTER IS INPUT FROM THE KEYBOARD. THE OUTPUT THEN IS THE DA"S VALUE AND A SINGLE CHARACTER STRING CORRESPONDING TO THE INPUT FROM THE KEYBOARD. WHILE THE TRACKING IS IN PROCESS, THE GRAPHIC POINTER IS DISPLAYED ON THE SCREEN. QCROSS THEN JUMPS TO GINSET FOR THE COMPLETION OF THE #POINTER# COMMAND.

USES: NO PSEUDO REGISTERS

CALLS: CROS, DIMSTR, ADRDEV, A7X, GINSET

RESTRICTIONS, ERRORS, NOTES: IF THE STRING VARIABLE HAS NOT BEEN PREVIOUSLY DIMENSIONED, DIMSTR CAN RETURN A MEMORY FULL ERROR.

REAHDR

FUNCTION: READS <READ> A 2-BYTE BINARY HEADER FROM THE CURRENT I/O DEVICE AND CHECKS IT FOR VALIDITY.

INPUTS: NONE

OUTPUTS: R0 : CONTAINS LENGTH INFORMATION IF IT WAS A STRING HEADER.

ACCA : 0 = VALUE HEADER
1 = STRING HEADER
-1,-2 = ILLEGAL HEADER
-3 = I/O ERROR OR EOF.

ERRCD : SET IF ERROR.

USES: R0

CALLS: BINBYT
BININ

NOTES: REQUIRES ALL NORMAL I/O SYSTEM VARIABLES BE SET UP CORRECTLY.

REASTG

FUNCTION: READS <READ> A STRING FROM THE CURRENT I/O DEVICE.

INPUTS: LENGTH : DIMENSIONED STRING LENGTH
POINT : POINTER TO LOCATION OF RESULT STRING.

OUTPUTS: 1)CHRCNT : NEW RESULT STRING LENGTH.
2) THE STRING IS STORED IN MEMORY STARTING AT POINT.
3) ERRCD IS SET IF ERROR OCCURED.

USES: R0
A,PTR
A,MAX
SCRTCH SPACE <SCRTCH=SCRTCH+255,>

CALLS: READHDR
BININ

NOTES: ALL NORMAL I/O SYSTEM VARIABLES MUST BE SET UP CORRECTLY.

REAVAL

FUNCTION: READ <READ> A VALUE FROM THE CURRENT I/O DEVICE.

INPUTS: POINT - POINTS TO LOCATION THAT IS TO RECEIVE THE VALUE.

OUTPUTS: READ VALUE.
ERRCD SET IF ERROR
EOF SET IF EOF ENCOUNTERED

USES: A,MAX
A,PTR

CALLS: A7X
BININ

NOTES: REQUIRES THAT ALL NORMAL I/O SYSTEM CONTROL VARIABLES BE SET UP CORRECTLY.

RENUM

FUNCTION: RENUM IS USED TO RENUMBER A PROGRAM POINTED TO BY PGMPTR.

INPUTS: R0=STARTING NUMBER
R1=INCREMENT VALUE
R2=STARTING STATEMENT NUMBER

OUTPUTS: A RENUMBERED PROGRAM IN ACCORDANCE WITH THE INPUTS.

USES: R0,R1,R2,R3,R4,R6,R7

CALLS: PSHRET, DISBLE, ENABLE, RTRN, PUSHX, GETLAR, GETSMA, A9X, INCXN, A8X

ERROPS: SAME AS FOR RENUMB

RENUMB

FUNCTION: RENUMB IS INVOLKED AS A RESULT OF THE BASIC #RENUMBER# COMMAND; IT IS USED TO RENUMBER A PROGRAM POINTED TO BY PGMPTR.

INPUTS: IT ACCEPTS UP TO THREE VALUES ON THE STACK AND SETS DEFAULTS FOR VALUES NOT SEEN AS FOLLOWS:

STARTING NUMBER = 100
INCREMENT VALUE = 10
STARTING STATEMENT NUMBER = 100

OUTPUTS: THE OUTPUT IS ESSENTIALLY A RENUMBERED PROGRAM FOR A PROGRAM THAT HAS NO RENUMBERABLE STATEMENT NUMBERS, RENUMB IS THE SAME AS A NO-OP.

USES: R0,R1,R2,R3,R4

CALLS: PSHRET, RENUM, TYPARG, FIX1

NOTES AND ERRORS: FOR A SUCCESSFUL RENUMBER, THE STACK IS LEFT CLEAN.

SINCE RENUMB MUST MODIFY CRITICAL AREAS OF A USERS PROGRAM, THROUGH RENUM, BREAKS AND ABORTS ARE DISABLED FROM TIME TO TIME DURING THE RENUMBER PROCESS.

ERRORS WILL RESULT UNDER THE FOLLOWING CONDITIONS:

1. IF ANY VALUE IS OUTSIDE, OR IS INCREMENTED OUTSIDE, THE RANGE OF 0-65,535.
2. IF THE STARTING NUMBER OR INCREMENT VALUE IS 0.
3. IF THE INITIAL PARAMETERS ARE SUCH THAT STATEMENT REPLACEMENT OR INTERLACING WILL OCCUR.

QRND

FUNCTION: GENERATES RANDOM NUMBERS WITH A UNIFORM DISTRIBUTION IN (0,1).

IF THE INPUT > 0 OR <=1> ANOTHER RANDOM NUMBER IS GENERATED (AND REPLACES THE KERNEL).

IF THE INPUT=0 THE KERNEL IS RESET TO A RANDOM VALUE.

IF 0> INPUT > -1, THE KERNEL IS RESET TO THE INPUT.

INPUT: FLOATING POINT NUMBER POINTED TO BY IX.

OUTPUT: FP NUMBER ON STACK

USES: KERNEL
R4,R7
T1,T2

CALLS: PSHRET
SHMR
PULFPN
SETRND
A9X
PSHFRN
INTMLC
INTMLA
NORM

NOTES: THE NEXT KERNEL (AND OUTPUT) IS GENERATED BY MULTIPLYING THE OLD KERNEL #5'17 MODULE 2'48.

RNDATA

FUNCTION: ROUNDS A FLOATING POINT NUMBER AND CONVERTS TO ASCII.

INPUTS: ACC A : NUMBER OF OUTPUT DIGITS DESIRED

X5...X0: FP MANTISSA, NOT NECESSARILY NORMALIZED.

DEXP, DEXP+1: POWER OF TEN WHICH MULTILIES THE MANTISSA.

OUTPUTS: N12,N11...N1 12 DIGIT ASCII FIELD

ES: EXPONENT SIGN, + OR - ASCII

E3,E2,E1: 3 DIGIT ASCII EXPONENT FIELD

USES: T4,T1

CALLS: TMULT

NOTE: 40 BASE 10 IS ADDED TO THE LAST BYTE OF THE MANTISSA PRIOR TO EXTRACTION OF THE NEXT DIGIT, THIS SOMETIMES CAUSES ROUNDING.

RPN

FUNCTION: RPN IS INVOLKED WHENEVER DIMENSIONING OR SUBSCRIPTING IS REQUIRED. ITS BASIC FLOW IS AS FOLLOWS:

FLOWCHART

SINCE THE PROCESSES ARE ALL QUITE DIFFERENT, A SEPERATE #INPUTS THRU NOTES# SECTION WILL BE PROVIDED FOR EACH PROCESS-

1) FRONT-END PROCESSING (FEP)
2) SUBSCRIPTING (SUB)
3) NEW ARRAY DIMENSIONING (NAD)
4) ARRAY RE-DIMENSIONING (ARD)
5) NEW STRING DIMENSIONING (NSD)
6) STRING RE-DIMENSIONING (SRD)

FEP INPUTS: BRKCNT AND DIMFLG ARE USED TO DETERMINE WHETHER DIMENSIONING OR SUBSCRIPTING IS TAKING PLACE.

ONE VALUE MUST BE ON THE STACK AND AN OPTIONAL SECOND VALUE MAY BE THERE; THE VALUE(S) ARE FOLLOWED BY A POINTER TO THE NAME TABLE.

IF ONLY ONE VALUE IS ON THE STACK, IT IS TAKEN AS THE ROW VALUE; IF TWO VALUES ARE ON THE STACK, THE FIRST VALUE POPPED IS THE COLUMN VALUE.

FEP OUTPUTS: INT1=ROW VALUE/STRING LENGTH

INT2= COLUMN VALUE (IF PRESENT).

DIMCNT= IS NON-ZERO IF INT2 IS VALID; OTHERWISE 0.

TPOINT= ABSOLUTE ADDRESS OF NAME TABLE ENTRY.

"X" REGISTER= ABSOLUTE ADDRESS OF NAME TABLE ENTRY.

"A" REGISTER= NAME TABLE POINTER STACK TAG.

FEP USES: R0,R4

FEP CALLS: PSHRET, TYPARG, FIX1

FEP ERRORS: AN ERROR WILL OCCUR IF THE VALUE(S) ARE NOT WITHIN THE RANGE 1<= V <= 65,535.

SUB INPUTS: SAME AS FRONT-END OUTPUTS.

SUB OUTPUT: PAETG ENTRY ON THE STACK (ABSOLUTE POINTER TO THE ARRAY ELEMENT PLUS SOME ASSUNDRY INFORMATION).

RPN (continued)

SUB USES: R7

SUB CALLS: INTMLA

SUB ERRORS: ERRORS OCCUR FOR THE FOLLOWING REASONS:

1)A SIMPLE VARIABLE HAS BEEN SUBSCRIPTED.
2) THE ROW VALUE IS OUT OF THE DIMENSIONED RANGE.
3) THE COLUMN VALUE IS OUT OF SYNC OR OUT OF RANGE WITH THE DIMENSIONED PARAMETERS.

NAD INPUTS: SAME AS FEP OUTPUTS.

NAD OUTPUTS: MEMORY IS ALLOCATED BY CREATING A NEW DATA ENTRY AS PER THE ROW AND COLUMN VALUES (DEFAULT COLUMN VALUE=1). THE ARRAY PARAMETERS ARE SET IN THE NAME TABLE. THE UNDEFINED BIT IS SET IN ALL VALUES.

NAD USES: R7

NAD CALLS: INTMLA, ARX, COMPR, DISBLE, PSHRET, ENABLE

NAD NOTES/ERROR: PART OF THE ALLOCATION ROUTINE MAY INVOLVE CONSIDERABLE TIME SINCE THE MEMORY COMPRESS ROUTINE MAY BE INVOLKED, ALSO, DURING THE TIME ACTUAL ALLOCATION IS TAKING PLACE, HARDWARE INTERRUPTS ARE DISABLED. IN ADDITION, DISBLE IS INVOLKED FOR BOTH THE ALLOCATION PROCESS AND DURING THE TIME THE UNDEFINED BITS ARE BEING SET IN THE VALUES.

ERRORS WILL RESULT IF THE USER ATTEMPTS TO DIMENSION A SIMPLE DEFINED VARIABLE OR IF THERE IS NOT ENOUGH MEMORY TO DO THE DIMENSIONING.

ARD INPUTS: SAME AS FEP OUTPUTS

ARD OUTPUTS: THE NEW ROW AND COLUMN VALUES ARE PUT IN THE NAME TABLE AND THE ALL OK BIT IS CLEARED.

ARD USES: R7

ARD CALLS: INTMLA, PSHRET

ARD NOTES/ERRORS: HARDWARE INTERRUPTS ARE DISABLED DURING THE SHORT TIME NEW VALUES ARE BEING UPDATED IN THE NAME TABLE.

AN ERROR WILL OCCUR IF THE USER TRIES TO REDIMENSION THE ARRAY IN SUCH A WAY AS TO INCREASE THE AMOUNT OF MEMORY ORIGINALLY ALLOCATED.

NSD INPUTS: SAME AS FEP OUTPUTS

NSD OUTPUTS: MEMORY IS ALLOCATED BY CREATING A NEW DATA ENTRY AS PER THE LENGTH VALUE. THE STRING PARAMETERS ARE SET IN THE NAME TABLE.

RPN (continued)

NSD USES: <=

NSD CALLS: PSHRET, DIMSTR

NSD NOTES: DIMSTR IS WHAT DOES MOST OF THE WORK FOR NAD.

SRD INPUTS: SAME AS FEP OUTPUTS

SRD OUTPUTS: THE NEW LENGTH IS PUT INTO THE NAME TABLE AND THE WORKING LENGTH IS SET EQUAL TO NEW LENGTH IF NEW LENGTH IS SHORTER THAN CURRENT WORKING LENGTH.

SRD USES: <=

SRD CALLS: PSHRET, DISBLE, ENABLE

SRD NOTES: AN ERROR WILL OCCUR IF THE USER TRIES TO REDIMENSION THE STRING IN SUCH A WAY AS TO INCREASE THE AMOUNT OF MEMORY ORIGINALLY ALLOCATED.

RTRN

FUNCTION: PULLS ONE SAVED RETURN ADDRESS OFF OF SECONDARY STACK AND PASSES CONTROL TO THAT POINT.

RESTRICTION, NOTES: YOU SHOULD JUMP TO THIS ROUTINE...

SAFE

FUNCTION: THIS ROUTINE IS USED TO LET ROM PACKS SERVICE INTERRUPTS WHEN THE SYSTEM MAY NOT BE SAFE. THE INTERRUPT ROUTINES WILL BE CALLED WHEN SAFE IS CALLED BUT THE SYSTEM INTERRUPT COUNTER IS NOT AFFECTED. SAFE SHOULD ONLY BE CALLED IF ALL SYSTEM POINTERS AND TABLES ARE IN CORRECT FORM. THE ABORT FUNCTION CAN NOT BE INITIATED DURING THE TIME SAFE IS ACTIVE.

RESTRICTION, NOTES: THIS ROUTINE CAN CALL ALL INTERRUPT PROCESSORS SO A LIST OF VARIABLES AND ROUTINES CAN NOT BE GENERATED. SEE ALSO DISBLE AND ENABLE.

SETRND

FUNCTION: SETS THE KERNEL OF THE RANDOM NUMBER GENERATER TO ITS DEFAULT VALUE.

OUTPUT: THE GLOBAL KERNEL IS SET TO ITS DEFAULT.

CALLS: PSHFPN
PULFPN

SHMR

FUNCTION: SHIFT MANTISSA RIGHT IN PLACE.

INPUT: ACC A CONTAINS THE NUMBER OF SHIFTS REQUIRED. IT IS DESTROYED. THE FP MANTISSA IS IN IX+3...IX+8.

OUTPUT: THE SHIFTED FP MANTISSA IS IN IX+3...IX+8.

SIG

FUNCTION: SIGNUM

INPUT: FP NUMBER X ON TOP OF STACK.

OUTPUT: FP 0.0 IF ON STACK X = 0
FP +1.0 IF ON STACK X > 0
FP -1.0 IF ON STACK X < 0

SLN

FUNCTION: LEFT SHIFT MANTISSA

INPUT: ACC A :NUMBER OF BIT POSITIONS TO SHIFT

IX : FP MANTISSA IS IN IX+3...IX+8

OUTPUT: ACC B : CONTAINS BITS SHIFTED OUT OF MANTISSA

IX+3...IX+8 CONTAIN THE REST OF THE SHIFTED MANTISSA.

SQR

FUNCTION: SQUARE ROOT

INPUT: FLOATING POINT NUMBER X ON STACK

OUTPUT: SQR(ABS(X)) ON STACK
EPSQR IS PLACED IN ERRCD IF X<0.0

USES: T1,R1

CALLS: PSHRET
DOFP
RTRN

NOTE: STARTS BY GETTING SQUARE ROOT TO 7 BITS BY THE DIVISION= LIKE AL

STKBLD

FUNCTION: BUILDS A #NORMAL FIX# STACK OF POINTERS OUT OF A #POST FIX# STACK ENTRY. FOR EXAMPLE: PRINT 1,2,3 LOOKS LIKE 3,2,1 ON THE STACK.

INPUT: R0 = POINTS TO BEGINNING OF #POST FIX# STACK

USES: DREXTB+1,+2

CALLS: CLRARG
DREXTB

NOTES: CLRARG IS USED TO GET NEXT TAG ON POST FIX STACK. IF THAT TAG IS AN EOL OR AN ATSNTG THE ROUTINE EXITS. IF A SEMITG IS ENCOUNTERED A SEMITG IS PUSHED ON THE #NORMAL# STACK. ALL OTHER ITEMS ARE TAGGED AS BAKSTG AND INCLUDES THE RESULTING BYTE FROM CLRARG AND AN INTERSTACK POINTER THAT POINTS TO THE ORIGINAL ENTRY IN THE POST FIX STACK. IT ALSO SKIPS ITM2TG WHICH IS USUALLY USED BY THE I/O SYSTEM AS ITS OWN EOL TAG.

SYMTAB

FUNCTION(S): SEARCHES THE SYMBOL TABLE FOR THE ENTRY CORRESPONDING TO THE SPECIFIED NAME. IF THE ENTRY IS FOUND A POINTER TO IT IS RETURNED; IF THE ENTRY IS NOT FOUND ONE IS CREATED.

INPUTS: PUT NAME IN R3 (TWO BYTES)
SET DEFFLG (BIT 'H20 IN TFLGS1) TO 0
SET STR (STRING)(BIT 'H10 IN TFLGS2) TO 0 OR 1
AS DESIRED, 1 IMPLIES STRING.

NOTE: TFLGS1 SHOULD BE RESTORED TO ITS ORIGINAL STATE IF THE TRANSLATOR WAS (MIGHT HAVE BEEN) ACTIVE WHEN THE ROM PACK GAINED CONTROL.

OUTPUTS: RETURNS A POINTER TO THE SYMBOL TABLE (NAME TABLE OR NT) IN R3

USES: R0, R3, R7, AND R8

CALLS: MAY CALL COMPR

RESTRICTIONS, SUGGESTIONS, NOTES: 1)THIS ROUTINE MAY CALL COMPR IF IT NEEDS TO CREATE A NEW ENTRY

2)USER NAMES ARE OF THE FORM
<LETTER><ASCII BLANK> OR
<LETTER><DIGIT> OR
<LETTER> &

NONE OF THESE FORMS SHOULD NORMALLY BE USED BY A ROM PACK TO AVOID COLLISIONS.

3)BECAUSE OF 1) AND THE NOTE PREVIOUS NEW ENTRIES FOR ROM PACKS SHOULD BE CREATED AT POWERUP TIME.

TMULT

FUNCTION: MULTIPLIES FP MANTISSA BY 10

INPUT: FP MANTISSA IN X5...X0, NEED NOT BE NORMALIZED.
ACC A, OFFSET, DESTROYED

OUTPUT: OVERFLOWED BITS INTO ACC B
REMAINDER OF MANTISSA IN X5...X0.
THE OFFSET THAT WAS IN ACC A IS ADDED TO THE REMAINDER FOR ROUNDING.

USES: Y0...Y5

TRIG

```
FUNCTION:       COMPUTES SIN, KOS AND TAN

INPUTS:         1. CONV1 POINTS TO DEGCON, GRDCON, OR RADCON ACCORDING
                TO THE TRIGONOMETRIC MODE

                2. THE ARGUMENT X IS ON THE STACK

                3. GLOBAL CTKN DETERMINES FUNCTION.

OUTPUTS:        THE RESULT IS RETURNED ON THE STACK

                1. IF THE ARGUMENT EXCEEDS ABOUT 40000 RADIANS A ZERO
                RESULT IS STACKED AND ERTPNG IS PLACED IN ERRC.

                2. IF TAN(X)=INFINITY THEN THE LARGEST FP NUMBER IS
                RETURNED; NO ERRCD IS SET.

APPROXIMATIONS  1. COS (X)=P(X'2) WHERE P, POLYNOMINAL OF DEGREE 6 IN
                X'2 WAS USED, CLAIMED TO BE A CHEBYSHEV APPROXIMANT,
                SIMILAR TO HART ?3823 WHICH HAS 16.25 DIGITS PRECISION

                2. SIN(X)=X*P(X'2) WHERE P IS OF DEGREE 5.  SIMILAR TO
                HART ?3044 WHICH IS GOOD TO 17.48 DIGITS.

                3. TAN(X)=X*P(X'2)/Q(X'2) WHERE P IS DEGREE 3 AND Q IS
                DEGREE 4.  SIMILAR BUT NOT THE SAME AS HART ?4286.

USES:           R1,R1+1
                T4

CALLS:          PSHRET
                PSHFPN
                FPMUL
                ZANS
                SLN
                NORMR
                FPSUB
                DOFP
                FPNEG
                RTRN

                FPDIV
```

TYPARG (TYPE OF ARGUMENT)

FUNCTION: TO TEST AND SIMPLIFY ARGUMENTS ON THE STACK, THIS ROUTINE USES R0 AS A PSEUDO STACK POINTER AND R4 TO RECORD THE RESULTS IT FINDS AND THE ACTIONS TAKEN. IF IT FINDS AN ARRAY IT WILL CALL MATSIZ TO INSPECT THE ARRAY FOR UNDEFINED ENTRIES AND TO PLACE THE END OF DATA ADDRESS ON THE STACK. MATSIZ ALSO PUTS THE SHAPE OF THE ARRAY ON THE STACK IN THE EXISTING STACK ENTRY. IF A LITERAL OR NAME TABLE STRING ENTRY IS FOUND IT IS CONVERTED TO A #POINTER TO STRING COUNT ENTRY.# IF A SIMPLE VARIABLE OR SUBSCRIPTED VARIABLE IS FOUND IT WILL BE CONVERTED TO A CONSTANT. THE ENTRIES FOR CONSTANTS AND POINTERS ARE ONLY NOTED IN R4 FOR THE CALLER. IF ANY STACK ENTRY IS FOUND FOR AN ARGUMENT R0 IS UPDATED TO POINT TO THE NEXT STACK ENTRY. IF THE END OF STACK IS FOUND R0 IS SET TO ZERO.

R4 BIT PATTERN MEANINGS:

RR AA BB CC

RR = BITS USED BY TTYPRES
AA = TYPE OF FIRST OPERAND.
BB = TYPE OF SECOND OPERAND.
CC = TYPE OF THIRD OPERAND.

BIT PATTERN MEANINGS:

00 = SCALER ENTRY.
01 = ARRAY ENTRY.
10 = STRING ENTRY.
11 = INVALID OPERAND. (NOTE ERUNDF MAY BE SET IN THIS CASE.)

INPUTS: R0 = A PSEUDO STACK POINTER, TAG ADDRESS=1.
R4 = HOLDING AREA ACCUMULATING INFORMATION ABOUT OPERANDS.

OUTPUTS: R0 = UPDATED IF A VALID STACK ENTRY WAS FOUND.
R4 = THE HIGH ORDER BYTE IS SHIFTED RIGHT TWO BITS AND INFORMATION ABOUT THE TYPE OF THE FOUND OPERAND IS PLACED IN BITS 4 AND 5.

USES: R0,R1, R2, R3, R4

CALLS: MATSIZ

RESTRICTIONS, NOTES: IF A POINTER TO STRING ENTRY IS ON THE STACK MEMORY COMPRESS CAN'T BE CALLED.

TYPE

FUNCTION: PREFORMS THE TYP(N) FUNCTION

INPUT: N = FILE NUMBER IS ON THE STACK.

OUTPUT: TYP (N) = IS RETURNED ON THE STACK.

USES: R0

CALLS: A9X
CRTRST
FIX 1
FLOAT 1
INITMT
MTOPEN
MTRBFR
PSHRET
RTRN
TYPFIL

TYPRES (TYPE OF RESULT)

FUNCTION: TEST FOR VALID RESULT AREA ON STACK. THIS ROUTINE WORKS IN CONJUNCTION WITH TYPARG TO TEST FOR A VALID STACK ENTRY TO STORE THE RESULT OF A FUNCTION. THE TYPE OF ITEM FOUND IS NOTED IN R4 FOR THE CALLER TO EXAMINE. IF AN ARRAY ENTRY IS FOUND THE END OF DATA ADDRESS AND SHAPE INFORMATION IS PLACED IN THE STACK ENTRY FOR THE ARRAY. IF A STRING IS FOUND AND IT IS NOT DIMENSIONED TYPRES WILL DIMENSION IT TO 72 BYTES.

INPUTS: R0 = A PSEUDO STACK POINTER (ADDRESS OF TAG-1).
R4 = WORK AREA FOR BIT PATERN DESCRIBING WHAT WAS FOUND.

OUTPUTS: R4 = BITS 6 AND 7 ARE SET TO DESCRIBE WHAT HAPPENED.

USES: R0, R4, AND LSP.

CALLS: MATSIZ, DISBLE AND ENABLE.

RESTRICTION, NOTES: IF TYPRES ALLOCATES A STRING, STACK SPACE IS USED.

UNADR

FUNCTION: RESET I/O SYSTEM AFTER AN I/O STATEMENT, GETS OFF OF GPIB.

INPUTS:
IOFUNC
A.STAT
A.PRIM
CRSTAT

OUTPUTS:
A.STAT=0
YAXIS=0
IOFLGS=0
PPMODE=0
NODUT=0
A.PRIM=255,
EOLCHR=15, <CR>
ISR = 65535, ; FOR ON PAGE

CALLS:
NEWBFR
ATNON
INTSRC
IFCSND
IECOFF
CRTRST

NOTE: IF DOING INPUT THEN THIS ROUTINE WILL SCAN FOR AN END OF RECORD IF ONE WAS NOT ALREADY ENCOUNTERED AS INDICATED BY CRSTAT.

UP

FUNCTION: A*B

INPUTS: B ON TOP OF STACK
A NEXT ON STACK

OUTPUT: A*B ON STACK UNLESS:

1. 0*0 = RESULT 0 AND ERRCD:= ERUP

2. 0*B = RESULT 0 (NO ERROR GIVEN, EVEN IF B<0\)

3. A*0 = RESULT 1 (NO ERROR)

4. IF B IS AN INTEGER <256
THEN A*ABS(B) IS COMPUTED BY FORMING POWERS OF A, E.G.
A, A*2, A*4, A*8, AND THEN MULTIPLYING THEM TOGETHER AS
NECESSARY. THEN IF B<0 THE INVERSE IS TAKEN. NO
CHECKS FOR OVERFLOW OR UNDERFLOW ARE MADE BY ERUP;
UNDERFLOWS WILL BE SET TO 0, OVERFLOWS TO THE LARGEST
REAL AND ERRCD:=ERFPOV

5. IF B IS NOT AN INTEGER <256 THEN
. 5.1 IF A<0 THEN BOTH OPERANDS ARE LEFT ON STACK AND
ERRCD:=ERUPN
. 5.2 OTHERWISE A*B = EXP (B*LOGE(A)). OVER/UNDERFLOWS
ARE HANDLED AND ZZETOX
. IF ANY OVERFLOW ERRORS ARE NOTED THEN
. ERRCD:=ERVP.

USES: R3
R1, R1+1

CALLS: PSHRET
A9X
PSHFPN
FPMUL
DOFP
RTRN
LOG
ZZETOX

UPCASE

FUNCTION: UPCASES ANY INPUT CHARACTER,
LOWER CASE A-Z TO UPPER CASE A-Z.
RIGHT BRACE TO)
LEFT BRACE TO (
RIGHT BRACKET TO)
LEFT BRACKET TO (

INPUTS: ACC-A HAS ASCII INPUT CODE.

OUTPUTS: ACC-B HAS ASCII OUTPUT CODE.

RESTRICION, NOTES: TESTCS IS A SPECIAL ENTRY POINT TO TEST THE STATE OF THE CASE/NOCASE SYSTEM STATUS AND EXIT OR FALL INTO UPCASE ACCORDINGLY.

VECTOR

FUNCTION: PUT VECTORS ON THE SCREEN

INPUT: TMP1:0 = DRAW
1 = MOVE

TMPHY: 2 BYTES <10 BITS> Y DATA = SCREEN POINTS

TMPHX: 2 BYTES <10 BITS> X DATA = SCREEN POINTS

CALLS: DRBUSY
WAIT.

NOTES: DISPLAY MUST BE ENABLED (NOT IN MAGTAPE MODE)

WRISTG

FUNCTION: WRITES <WRITE> A STRING ON TO THE CURRENT I/O DEVICE.

INPUTS: POINT : POINTS TO BEGINNING OF STRING
LENGTH : LENGTH OF STRING

OUTPUTS: ERRCD:SET IF ERROR

USES: R0

CALLS: BINOUT

NOTES: ALL NORMAL I/O SYSTEM VARIABLES MUST BE SET UP CORRECTLY.

WRIVAL

FUNCTION: WRITES <WRITE> A VALUE ONTO THE CURRENT I/O DIVICE

INPUTS: POINT : POINTS TO VALUE

OUTPUTS : ERRCD : SET IF ERROR IN DEVICE HANDLERS.

USES: A.PTR
A.END

CALLS: BINOUT
A7X

XFRCTL

FUNCTION: ATTACH AN I/O HANDLER FOR THE PRESENT A,PRIM DEVICE, WILL PERFORM SEARCH FOR DEVICE HANDLERS IN ROM PACKS.

INPUTS: A,PRIM ; PRIMARY ADDRESS

INDEX REG, ; POINTING TO A TABLE AS DEFINED BELOW

A,PRIM=0 - INDICATES FILE HANDLER

1-30 - INDICATES IEC BUS HANDLER

31 - INDICATES KEYBOARD HANDLER

32 - INDICATES DISPLAY HANDLER

33 - INDICATES MAGTAPE HANDLER

34 - INDICATES DATA STATEMENT HANDLER

35 - INDICATES 2ND MAGTAPE HANDLER

36-255, WILL RUN PIATBL FOR HANDLER,

OUTPUT: ERRCD ; SET IF CAN"T FIND DEVICE HANDLER OR DEVICE HANDLER CAUSES AN ERROR,

USES: BANK
PIATBL

CALLS: JMPX
NIODEV
A6X
SETBNK