BASIC CALLABLE SUBROUTINES/FUNCTIONS

Following are descriptions of various subroutines and functions available to BASIC programmers operating in the TODS environment.

CHAIN

PURPOSE

ähnlich wie GOSUB, nur, daß Sub-Routine auf Disc abgelegt ist.

Transfers control to another executable program on the disc.

FORMAT EXAMPLE

110 CHAIN (PRN)

Where PRN is the program reference number of an executable program on the disc.

COMMENTS

CHAIN may be used to transfer control to any executable program on the disc, whether written in BASIC, FORTRAN, ALGOL, or Assembler language. The effect of a CHAIN call is modified by TODS REPEAT and SINGLE PROGRAM commands as follows:

If a CHAIN is encountered in a program while TODS is operating in the Repeat Program Mode, the CHAIN is executed as a STOP. Execution of the program containing the CHAIN is started over at the beginning.

If the Single Program flag has been set, a CHAIN will cause TODS to go to the Stop/Ready state. The program called for by the CHAIN will be executed when the RUN/CONTINUE command is issued.

If a CHAIN is encountered in a program while TODS is operating simultaneously in the Repeat and Single Program Modes, it is executed as a STOP. TODS will go to the Stop/Ready state with the program containing the CHAIN in core. When the RUN/CONTINUE command is given, execution of the program containing the CHAIN is started over at the beginning.

CLOSE

PURPOSE

Terminates access to a disc file.

FORMAT EXAMPLE

510 CLOSE (L, E)

where:

L = disc logical unit number (7, 9, 10 or 11)
E = indicates status as follows:
0 = disc file closed
1 = disc file cannot be closed

NOTE

The E (error) parameter is not checked by TODS or BASIC, only returned for an indication of status. To avoid loss of data, the test programmer must provide for a status check in the calling program upon return from this call.

COMMENTS

CLOSE causes the file position pointers – maintained in core during the time the file is open – to be written into the catalog for the file. These pointers are re-established when the file is opened again. CLOSE should be used following a sequential DRITE in order to preserve the data. Close should normally not be used following a DREAD.

CRTCL

PURPOSE

Clears the screen on the CRT terminal and returns the cursor to the upper left-hand corner of the screen.

FORMAT EXAMPLE CRT - Clear

25 CRTCL

COMMENTS

No parameters are required for this call. On systems that do not include a CRT terminal, this call will have no effect.

DREAD/DRITE

PURPOSE

Reads/writes an array of floating point variables (two words per variable) on the disc.

FORMAT EXAMPLE

10 DREAD (L, F, C, N, E) – Read
20 DRITE (L, F, C, N, E) – write

where:

L = logical disc unit (7, 9,10 or 11) on which a floating point array (2 words per element) is to be read or written

F = 0, a sequential read/write in which the beginning ordinal location is governed by the disc read/write pointers

or:

F = positive integer, indicating a random access read/write and pointing to the ordinal within the disc file of the first floating point variable to be read from or written to (beginning word = 2 (F–1) + read/write pointer)

C = indicates core location of first variable to be transferred, e.g. A(1)

N = integer, indicating number of variables to be transferred

E = indicates status as follows: wird v. Betriebssystem gesetzt.

0 = no error
1 = illegal LU
2 = read or write attempted outside of file limits or attempt to read unwritten data

NOTE

The E (error) parameter is not checked by TODS or BASIC, only returned for an indication of status. To avoid loss of data, the test programmer must provide for a status check in the calling program upon return from this call.

COMMENTS

Two access modes are available: sequential – where N floating point variables are transferred from the current position of the disc file pointers (causing the pointers to be updated); and random access – where the disc file pointers are not used or altered.

Data is transferred to or from the disc file assigned to L by the OPEN function. BASIC checks for validity of transfer data; if not valid, it aborts the operation and sets E = 1.0. Other call parameter errors are: negative L, F, or N or N = 0.

NOTE

Unlike most BASIC callable drivers DREAD/DRITE returns control to the calling program without printing any error messages.

EPRS

Extended - Program Request - State = CRTL A

PURPOSE

Causes TODS to go to the Program Request state.

FORMAT EXAMPLE

175 EPRS

FAIL

PURPOSE

Pass und Fail - Lampen am Control - Panel können angesprochen werden

Permits a program to light or extinguish the FAIL/NO lamp on the control panel.

FORMAT EXAMPLE

10 FAIL (X)

where:

X = 1, to light FAIL lamp and extinguish PASS lamp
= 0, to extinguish FAIL lamp

INPUT

PURPOSE

Causes TODS to go to the Input state and lights the INPUT lamp on the control panel.

FORMAT EXAMPLE

```
10 INPUT X
```

where:

X = variable in BASIC program to be assigned input value

COMMENTS

Values may be input through the control panel keyboard or the system terminal.

After entering the requested value, press the INPUT button on the control panel or RETURN key on the system terminal. The INPUT lamp is then extinguished, and the value input is transferred to the program.

NOTE

When entering negative values, the minus sign must be the first key pressed. Pressing the (–) key after a numeric entry will change the VALUE/LINE NUMBER display to show a negative number, but the value recognized by TODS will be positive. For example, the key sequence: 1, 2, 3, –, INPUT will display "– 1 2 3"; but TODS will interpret this entry to be (+) 1 2 3.

INVOKE

PURPOSE

Transfers control to another executable program on the disc that is to be used as a subroutine.

FORMAT EXAMPLE

```
75 INVOKE (PRN)
```

where:

IPRN is the program reference number of an executable program on the disc.

COMMENTS

INVOKE causes control to be transferred to the program specified by PRN. Following execution of PRN, the calling program is reloaded and control is transferred to the next sequential program statement following the INVOKE statement.

INVOKE may be used to transfer control to any executable program on the disc, whether written in BASIC, FORTRAN, ALGOL, or Assembler language. The effect of an INVOKE call is modified by TODS REPEAT and SINGLE PROGRAM commands as follows:

If an INVOKE is encountered in a program while TODS is operating in the Repeat Program Mode, the INVOKE is executed normally. Following execution of the program called by the INVOKE, execution of the main program is continued at the point following the INVOKE. Upon completion of the main program, the main program is repeated.

If the Single Program flag has been set, or an INVOKE is encountered in a program while TODS is operating simultaneously in the Repeat and Single Program Modes, TODS will go to the Stop/Ready state with the program called by INVOKE in core. This "subroutine" program is executed when the RUN/CONTINUE command is given. Following execution of the subroutine program, the remainder of the main program is executed without another pause. TODS will return to the Stop/Ready state upon completion of the main program.

LPPAGE

PURPOSE

Causes the line printer to skip to the top of the next page.

FORMAT EXAMPLE

45 LPPAGE

COMMENTS

No parameters are required for this call. This call will have no effect on systems that do not include a line printer.

OPEN

PURPOSE

Permits access to a disc file by associating a disc file reference number with a disc logical unit number.

FORMAT EXAMPLE

10 OPEN (L, F, E)

where:

L = disc logical unit number (7, 9, 10 or 11)

F = file reference number to be associated with LU

E = indicates status as follows:
0 = disc file open
1 = disc file cannot be opened

NOTE

The E (error) parameter is not checked by TODS or BASIC, only returned for an indication of status. To avoid loss of data, the test programmer must provide for a status check in the calling program upon return from this call.

COMMENTS

A call to OPEN must precede any initial disc read or write operation from a user's program. OPEN need be used only once to enable access to a particular file on the disc, i.e., following a call to OPEN, any number of read or write operations may be performed on the disc file specified by F without calling OPEN prior to each operation.

OPNUM

PURPOSE

Permits a program to display a number in the OPERATION window on the control panel.

FORMAT EXAMPLE

10 OPNUM (I)

where:

I = number to be displayed; range: 0 through 99

COMMENTS

If $99 < I \leqslant 32767$, only the last two significant digits of I are displayed. If $0 > I > 32767$, the display is cleared.

PASS

PURPOSE

Pass - Lampe k. aufgespr. werden

Permits a program to light or extinguish the PASS/YES lamp on the control panel.

FORMAT EXAMPLE

10 PASS (X)

where:

X = 1, to light PASS lamp and extinguish FAIL lamp
= 0, to extinguish PASS lamp

PAUSE

PURPOSE

Causes TODS to go to the Pause state and lights the PAUSE lamp on the control panel.

FORMAT EXAMPLE

```
10 PAUSE
```

COMMENTS

Program execution may be continued by issuing the RUN/CONTINUE command.

RESET

PURPOSE

Resets disc file read or write pointers on a specified logical disc unit.

FORMAT EXAMPLE

```
210 RESET (L, I)
```

where:

L = logical disc unit (7, 9, 10 or 11)

I = 0, reset read and write pointers on L
= 1, reset read pointers only
= 2, reset write pointers only

COMMENTS

Resetting the write pointers effectively marks that file as containing no data.

wenn jetzt 5 Daten gesch. werden, so kann m. Reset wieder bei 0 angefangen werden

SCOM

PURPOSE

Provides access to system common storage.

FORMAT EXAMPLE

10 SCOM (M, V, C, E)

where:

M = mode
0 = read
1 = write

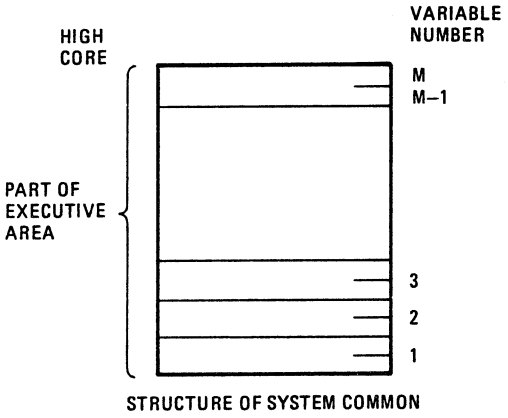
V = variable in user's program

C = ordinal of variable in system common area; $1 \leqslant C \leqslant 20$

E = 0, no error
= 1, index out of range

COMMENTS

V and C can be variables or constants.



STRUCTURE OF SYSTEM COMMON

SUPV (X)

Control-Panel-Switch kann abgefragt werden

PURPOSE

Permits a program to check the position of the control panel OPERATOR/SUPERVISOR key switch.

FORMAT EXAMPLES

```
10 IF SUPV (X) GOTO LLLL
   or
20 LET A = SUPV (X)
```

where:

SUPV (X) = 1, if the OPERATOR/SUPERVISOR key switch is in the SUPERVISOR position

SUPV (X) = 0, if the OPERATOR/SUPERVISOR key switch is in the OPERATOR position

The X in SUPV (X) is an unused dummy variable

LLLL = a BASIC program line number

COMMENTS

When a statement of the form: IF SUPV (X) GOTO LLLL is encountered in a BASIC program, and SUPV (X) = 1, control is transferred to line number LLLL. If SUPV (X) = 0, control is transferred to the next statement of the program.

TSNUM

PURPOSE

Permits a program to display a number in the TEST NUMBER window on the control panel.

FORMAT EXAMPLE

10 TSNUM (X)

where:

X = number to be displayed; range: 0 through 99999

VLNUM

PURPOSE

Permits a program to display a number in the VALUE/LINE NUMBER window on the control panel.

FORMAT EXAMPLE

10 VLNUM (X)

where:

X = floating point number to be displayed; range: plus or minus 0.000001 through plus or minus 999999 and 0

COMMENT

In Supervisor Mode the VLNUM call is of little use since TODS uses the VALUE/LINE NUMBER window to display the line number of the current BASIC statement being executed. Any value displayed would be immediately replaced by the line number of the next statement in the program.

While no harm is done by executing this call in Supervisor Mode, some execution time can be saved by limiting its use to Operator Mode only by means of this preferred format example:

10 IF SUPV(X)=0 VLNUM(V)

The SUPV(X) function is used to check operating mode; VLNUM(V) is executed only if TODS is in Operator Mode.

YES (0)

PURPOSE

Causes TODS to go to the Yes/No state and permits a program to ascertain if the PASS/YES or the FAIL/NO button on the control panel has been pressed.

FORMAT EXAMPLES

```
10 IF YES (0) GOTO X
   or
20 LET P = YES (0)
```

where:

YES (0) = 1, if the PASS/YES button on the control panel or the Y key on the system terminal keyboard is pressed; control is transferred to statement number specified by X

YES (0) = 0, if the FAIL/NO button on the control panel or the N key on the system terminal keyboard is pressed; control is transferred to the next statement of the program

COMMENTS

When a statement of the form: IF YES (0) GOTO X is encountered in a BASIC program, the PASS/YES and FAIL/NO lamps on the control panel are lighted and the program waits for operator response before continuing execution. When the operator has replied by pressing the PASS/YES or FAIL/NO button, both lamps are extinguished and program execution continues.

Refer also to the FAIL and PASS call descriptions in this section for details on lighting and extinguishing the PASS/YES and FAIL/NO button lamps under program control.

APPENDIX B
SUMMARY OF COMMANDS

INTRODUCTION

This appendix contains a summary of TODS system commands, ATS BASIC commands, Librarian commands, and TODS Editor commands. For more detailed descriptions of these commands, refer to the appropriate section of this manual.

SYSTEM COMMANDS

Command	Purpose	Format
PROGRAM REQUEST	Returns TODS to the Program Request state.	Press LOAD button on control panel or Control-A on system terminal.
STOP/READY	Sets TODS to the Stop/Ready state with the next job stack program in core.	Press READY/STOP button on control panel or Control-S on system terminal.
RUN/CONTINUE	Initiates program/job stack execution or continues execution from the Pause state.	Press RUN/CONTINUE button on control panel or Control-R on system terminal.
PAUSE	Suspends execution of the current program in core until the RUN/CONTINUE command is issued.	Press PAUSE button on control panel or Control-P on system terminal. (Recognized only in Supervisor Mode.)
YES/NO	Indicates yes or no response from operator.	Press PASS/YES or FAIL/NO button on control panel or Y or N key on system terminal. RETURN key is not required.
TRAP 1	Sets the status bit for a type 1 interrupt during execution of a BASIC program.	Press TRAP 1 button on control panel or Control-Q on system terminal.
TRAP 2	Sets the status bit for a type 2 interrupt during execution of a BASIC program.	Press TRAP 2 button on control panel or Control-W on system terminal.
SINGLE PROGRAM	Causes TODS to go to the Stop/Ready state with the next job stack program in core, following execution of each program in the job stack.	Set switch 6 up on control panel SWITCH REGISTER.

SYSTEM COMMANDS (Continued)

Command	Purpose	Format
REPEAT PROGRAM	Causes TODS to continuously execute program in core.	Set switch 7 up on control panel SWITCH REGISTER.
RUBOUT	Causes TODS to ignore the current line of input.	Press CLR key on control panel or RUBOUT or DEL key on system terminal.
END-OF-RECORD	Indicates to TODS that the current keyboard or control panel entry is complete.	Press INPUT button on control panel or RETURN key on system terminal.
CONTROL-^	Stops input processing of underscore or back arrow as indicating a backspace to delete the immediately preceding character(s) in a line of input.	Press Control-^ or Control-↑ on system terminal. (Recognized only in Supervisor Mode.)
CONTROL-_	Restores input processing of underscore or back arrow as indicating a backspace to delete the immediately preceding character(s) in a line of input.	Press Control-_ or Control-← on system terminal. (Recognized only in Supervisor Mode.)

ATS BASIC COMMANDS

Command	Purpose	Format
CATALOG	Lists the directory of all BASIC programs currently stored on the disc.	Enter CATALOG (Similar to Librarian FNLIST)
DELETE	Deletes the BASIC program, or segments of the BASIC program, currently in core.	Enter DEL or DELETE DEL k1 DELETE k1 DEL k1,k2 DELETE k1,k2
DLOAD	Loads a BASIC program from the disc into core.	Enter DLOAD prn
DSAVE	Stores the BASIC program currently in core on the disc.	Enter DSAVE prn DSAVE prn, name DSAVE prn, crn, name
DSPLAY	Displays a BASIC program on the system display device.	Enter DSP or DSPLAY DSP k1 DSPLAY k1 DSP k1,k2 DSPLAY k1,k2
LIST	Lists a BASIC program on the system list device.	Enter LIST LIST k1 LIST k1,k2

ATS BASIC COMMANDS (Continued)

Command	Purpose	Format
LOAD (N)	Loads a BASIC source program into core from the system load device (usually punched tape reader) or the disc.	Enter LOAD (if from input device) or LOAD srn (if from disc)
PAGE	Issues a page eject command to line printer.	Enter PAGE
REMOVE	Marks the indicated BASIC program (stored on the disc) as inactive.	Enter REMOVE prn (Similar to Librarian DELETE)
REPLACE	Replaces the indicated BASIC program on the disc with the BASIC program currently in core.	Enter REPLACE REPLACE prn REPLACE prn, name REPLACE prn, crn, name
RUN/CONTINUE	Initiates execution of the BASIC program currently in core, or continues execution of a BASIC program from the Pause state.	Press RUN/CONTINUE button on control panel or Control-R on system terminal.
SAVE	Stores a BASIC source program on paper tape.	Enter SAVE SAVE k1 SAVE k1,k2
SSAVE	Stores a BASIC source program on the disc.	Enter SSAVE srn SSAVE srn, name SSAVE srn, crn, name
STOP	Terminates program execution.	Press READY/STOP button on control panel or Control-S on system terminal.

APPENDIX C
QUICK REFERENCE TO BASIC

OPERATORS

SYMBOL	SAMPLE STATEMENT	PURPOSE/MEANING/TYPE
=	11Ø LET A = Ø 115 B = 5	Assignment operator; assigns a value to a variable.
↑	12Ø PRINT X↑2	Exponent (as in X^2).
*	13Ø LET C5 = (A*B)*N2	Multiply
/	14Ø PRINT T5/4	Divide
+	15Ø LET P = R1 + 1Ø	Add
-	16Ø X3 = R3 - P	Subtract
	NOTE: The numeric values used in logical evaluation are: true = any non-zero number; false = Ø.	
=	17Ø IF D=E THEN 6ØØ	*expression* equals *expression*
#	18Ø IF (D+E)#(2*D)THEN 71Ø	*expression* does not equal *expression*
< >	18Ø IF(D+E)< >(2*D)THEN 7ØØ	*expression* does not equal *expression*
>	19Ø IF X>1Ø THEN 62Ø	*expression* is greater than *expression*
<	2ØØ IF R8<P7 THEN 64Ø	*expression* is less than *expression*
>=	21Ø IF R8>=P7 THEN 71Ø	*expression* is greater than or equal to *expression*
<=	22Ø IF X2<=1Ø THEN 65Ø	*expression* is less than or equal to *expression*
AND	23Ø IF G2 AND H5 THEN 9ØØ	*expression* 1 and *expression* 2 must both be true for statement to be true.
OR	24Ø IF G2 OR H5 THEN 91Ø	If *expression* 1 or *expression* 2 is true, statement is true.
NOT	25Ø IF NOT G5 THEN 95Ø	Statement is true when *expression* (NOT G5) is false.

LANGUAGE STATEMENTS

STATEMENT TYPE	EXAMPLE	PURPOSE
DATA	36Ø DATA 99,1Ø6.7, 16.2	Specifies data; read from left to right
DIM	31Ø DIM A(72)	Specifies maximum array size.
COM	1 COM A(1Ø)	Allows a BASIC program to store data in memory using subscripted variables for retrieval by a subsequent BASIC program and specifies maximum array size.
END	4ØØ END	Terminates program execution (optional).
FOR...NEXT	35Ø FOR J=1 TO N STEP 3 . . . 4ØØ NEXT J	Executes statements between FOR and NEXT the specified number of times and in increments indicated after STEP; STEP and STEP value may be omitted.
GO TO	33Ø GO TO 9ØØ	Transfers control to specified statement.
GOSUB	42Ø GOSUB 8ØØ	Begins executing the subroutine at specified statement (see RETURN).
IF...	34Ø IF A#1Ø PRINT "A>1Ø" 345 IF A<Ø GO TO 1ØØ	Logical test of specified condition; if true, the specified action is carried out.
INPUT	39Ø INPUT Y2,B4	Allows data to be entered from the input device.
ASSIGNMENT OR COMPUTATIONAL	25 LET X=A+B	Assigns the value of the expression on the right of the equal sign to the variable on the left of the equal sign.
	3Ø X=A=Ø	Multiple assignments are allowed. LET is optional.
NEXT	355 NEXT J	Sets the boundary of the FOR loop.
READ	36Ø READ A,B,C	Reads information from DATA statement.
REM	32Ø REM--ANY TEXT**!!	Inserts remarks in a program.
PRINT	356 PRINT A,B,"HELLO"	Tells computer to print information.

LANGUAGE STATEMENTS (Continued)

STATEMENT TYPE	EXAMPLE	PURPOSE
PRINT TAB(X)	357 PRINT TAB(2Ø),A	The print device line is divided into 72 columns. Insertion of TAB (X) will cause the print device to move to column X.
	358 PRINT	Causes print device to *line feed*.
RESTORE	38Ø RESTORE	Permits re-reading from DATA statement.
RETURN	85Ø RETURN	Transfers control to statement following associated GOSUB.
STOP	41Ø STOP	Terminates program execution.
PAUSE	6Ø PAUSE	Temporarily suspends program execution.
DSPLAY	7Ø DSPLAY X,Y,Z	Tells computer to display information on display device. See PRINT.
LOAD	8Ø LOAD X,Y,Z	Reads ASCII information into computer from load device.
SAVE	9Ø SAVE X,Y,Z	Outputs ASCII information from computer to record device. See PRINT.
WAIT	9ØØ WAIT (1ØØØ)	Introduces delays into a program. Causes the program to wait a specified number of milliseconds before continuing execution.
TRAP	6ØØ TRAP 3 GOSUB 6Ø	When the specified interrupt type occurs, go to the subroutine starting at the specified line number.
FAIL:	5Ø DVMRD (3,X,Y) FAIL: GOSUB 9000	Specifies action to be taken if an error is indicated on return from an assembly language subroutine call.

FUNCTIONS

NOTE: PRINT is used for examples only; other statement types may be used.

FUNCTION TYPE	EXAMPLE	PURPOSE
DEF FN	3ØØ DEF FNA (X)=(M*X)+B	Allows the programmer to define functions; the function label (FNA) must be a letter from A to Z; the argument (X) must be mentioned in the function definition.
ABS(X)	31Ø PRINT ABS (X)	Gives absolute value of expression (X).
EXP (X)	32Ø PRINT EXP (X)	Gives constant *e* raised to power of expression (X); in this example, *e*↑X.
INT (X)	33Ø PRINT INT (X)	Gives largest integer less than or equal to expression (X).
LOG (X)	34Ø PRINT LOG (X)	Gives natural logarithm of an expression; expression must have a positive value.
RND (X)	35Ø PRINT RND (X)	Generates a random number between Ø and 1; expression (X) is a dummy argument.
SQR (X)	36Ø PRINT SQR (X)	Gives square root of expression (X); expression must have a positive value.
SWR (X)	365 IF SWR (X) GO TO 7Ø	Gives logical value, 1 or Ø, of indicated SWITCH REGISTER switch position (X).
SIN (X)	37Ø PRINT SIN (X)	Gives sine of expression (X); X is real and in radians.
COS (X)	38Ø PRINT COS (X)	Gives cosine of expression (X); X is real and in radians.
TAN (X)	39Ø PRINT TAN (X)	Gives tangent of expression (X); X is real and in radians.
ATN (X)	4ØØ PRINT ATN (X)	Gives arctangent of expression (X) in radians (range is $-\pi/2$ to $+\pi/2$; X is real.
SGN (X)	440 PRINT SGN (X)	Gives: 1 if X>Ø, Ø if X=Ø, -1 if X<Ø.
IERR (X)	5ØØ IF IERR (X) GOSUB 9ØØØ	Returns value stored in error code cell by user program or subroutine.
SERR (X)	51Ø SERR (A)	Sets error code cell to value of A.

ARRAYS

NOTES: 1. Maximum array size is 255 elements.
2. Array variables must be a single letter from A to Z.

NAME	SAMPLE STATEMENT	PURPOSE
DIM	1Ø DIM A (1Ø, 2Ø)	Allocates space for an array of the specified dimensions.
INPUT	4Ø INPUT A(5,5)	Inputs specified array element from the input device.

APPENDIX D
ERROR CODES, MEANING AND PROBABLE CAUSE

INTRODUCTION

This appendix contains information pertaining to error messages printed by BASIC. Following is a list of error codes numbers not used in ATS BASIC: 4, 6, 8, 22, 23, 24, 25, 26, 27, 28, 29, 38, 44, 46, 58, 59, 60, 61.

ERROR CODE	MEANING	PROBABLE CAUSE
1	STATEMENT ENDS UNEXPECTEDLY.	Statement end found by the syntax analyzer. Additional characters are needed to form a complete statement.
2	EXCEEDS 72 CHARACTERS.	Too many characters in the line.
3	SYSTEM COMMAND NOT RECOGNIZED.	The line just typed begins with a letter, but the initial character string does not form a recognizable statement type. May be a missing statement number.
5	BAD EXPONENT.	A number appears followed by an E but not followed by a legitimate exponent integer.
7	LET STATEMENT HAS NO STORE.	No assignment operator appears in the formula following LET.
8	MULTIPLE OR MISPLACED COM STATEMENT.	A COM statement does not have the lowest line number, or is the second COM statement in the program.
9	MISSING OR INCORRECT FUNCTION IDENTIFIER IN DEF.	DEF is not followed by FN <letter>, or FN is not followed by a letter in a formula (for example, A+FN(3)).
10	MISSING PARAMETER IN DEF STATEMENT.	No simple variable is found following a DEF FN <letter>.
11	MISSING ASSIGNMENT OPERATOR.	No assignment operator found in a DEF or FOR statement.
12	MISSING OR INCORRECT STATEMENT TYPE AFTER IF.	An IF statement has an incorrect statement following the decision formula.
13	MISSING OR INCORRECT FOR-VARIABLE.	A simple variable is not found following a FOR or a NEXT; for example, FOR A(1) = ... is not legal.

ERROR CODES (Continued)

14	MISSING TO.	No TO found following the initial part of a FOR statement. May also be an incorrect formula as in FOR I = BC TO ..
15	INCORRECT STEP IN FOR STATEMENT.	Characters appear following the limit formula but do not form a correct STEP formula. May also be an incorrect formula as in FOR I = 1 to A+BC STEP Y (operator missing between B and C).
16	CALLED ROUTINE DOES NOT EXIST.	Subroutine not defined for this system.
17	WRONG NUMBER OF PARAMETERS IN SUBROUTINE CALL.	
18	MISSING OR INCORRECT CONSTANT IN DATA STATEMENT.	Caused by such things as: DATA 1,2, *return* (trailing comma) or DATA ++3 May also be caused by an incorrect reply to an INPUT statement, as +-4.
19	MISSING OR INCORRECT VARIABLE IN READ, INPUT, OR LOAD STATEMENT.	No variable appears following a READ, INPUT, or LOAD statement or there is a trailing comma following one of these statements; for example READ A,B,.
20	NO CLOSING QUOTE FOR PRINT STRING.	Unmatched " in a PRINT statement.
21	MISSING PRINT DELIMITER OR BAD PRINT QUANTITY.	Caused by such things as missing operators or commas. Generally means that two formulas appear to be concatenated without separating punctuation.
30	MISSING LEFT PARENTHESIS.	Causes should be obvious from inspection of the line.
31	MISSING RIGHT PARENTHESIS.	Cause should be obvious from inspection of the line.
32	OPERAND NOT RECOGNIZED.	No recognizable operand found where one is expected, for example, following a binary operator or left parenthesis. Usually a typing error.
33	DEFINED ARRAY MISSING SUBSCRIPT PART.	An array appearing in a DIM or COM statement does not have a proper subscript-bound part; for example, DIM A(3),B,C(4,5). (B has no subscript.)

ERROR CODES (Continued)

ERROR CODE	MEANING	PROBABLE CAUSE
34	MISSING ARRAY IDENTIFIER.	No array identifier found where one is expected in a DIM or COM statement.
35	MISSING OR BAD INTEGER.	A required integer is Ø, or too large, or does not appear at all. Required integers appear as sequence numbers, formal bounds in DIM and COM statements, and as the first parameter of a Call statement.
36	NON-BLANK CHARACTERS FOLLOWING STATEMENT'S LOGICAL END.	Extra characters have been found in a statement at a place where the statement should logically end. For example, 1ØØ RESTORE DATA. The word DATA should not be present.
37	OUT OF STORAGE DURING SYNTAX PHASE.	Program is too large.
39	DOUBLY DEFINED FUNCTION.	The same function is defined in two DEF statements.
40	FOR STATEMENT HAS NO MATCHING NEXT STATEMENT.	
41	NEXT STATEMENT HAS NO MATCHING FOR STATEMENT.	The program contains an extra NEXT statement, or an improper nesting of FOR -- NEXT loops.
42	OUT OF STORAGE FOR SYMBOL TABLE.	Program is too large.
43	ARRAY APPEARS WITH INCONSISTENT DIMENSIONS.	An array is referenced as being singly-subscripted in one place and as doubly-subscripted in another.
45	ARRAY DOUBLY DIMENSIONED.	The same array appears twice in DIM statements, or in both a DIM statement and a COM statement.
47	ARRAY TOO LARGE.	Array size exceeds limits of available memory.
48	OUT OF STORAGE DURING ARRAY ALLOCATION.	Program and arrays together are too large.

ERROR CODES (Continued)

ERROR CODE	MEANING	PROBABLE CAUSE
49	SUBSCRIPT EXCEEDS BOUND.	An actual subscript exceeds the bound declared in a DIM or COM statement, or exceeds 1Ø, if undeclared.
50	ACCESSED OPERAND HAS UNDEFINED VALUE.	Attempted use of a variable or array element which has never been assigned a value.
51	NON-INTEGER POWER OF NEGATIVE NUMBER.	
52	ZERO TO ZERO POWER.	
53	MISSING STATEMENT.	Attempted GO TO or GOSUB to a non-existent statement.
54	GOSUBS NESTED MORE THAN 2Ø DEEP.	Attempted execution of more than 2Ø GOSUBs in a row, with no intervening RETURNs. (Refers to the logical execution of a program, not the physical layout.)
55	RETURN FINDS NO ADDRESS.	RETURN is encountered during execution when no GOSUBs are active.
56	OUT OF DATA.	More data has been requested in a READ statement than exists in the DATA statement.
57	OUT OF STORAGE DURING EXECUTION.	Insufficient working space to execute the program.
62	TRIGONOMETRIC FUNCTION ARGUMENT IS TOO LARGE.	Applies to SIN, COS, and TAN.
63	ATTEMPTED SQUARE ROOT OF NEGATIVE ARGUMENT.	
64	ATTEMPTED LOG OF NEGATIVE ARGUMENT.	

ERROR CODES (Continued)

ERROR CODE	MEANING	PROBABLE CAUSE
	WARNING-ONLY ERRORS (Program continues executing.)	
65	NUMERICAL OVERFLOW, RESULT TAKEN TO BE + OR - INFINITY.	A calculated result or number input exceeds the capacity of numerical representation. The value is replaced by the largest representable number of appropriate sign.
66	NUMERICAL UNDERFLOW, RESULT TAKEN TO BE ZERO.	Number is too close to zero to be represented as other than zero.
67	LOG OF ZERO TAKEN TO BE -INFINITY	
68	EXP OVERFLOWS, RESULT TAKEN TO BE +INFINITY.	
69	DIVISION BY ZERO, RESULT TAKEN TO BE + OR -INFINITY.	
7Ø	ZERO RAISED TO NEGATIVE POWER, RESULT TAKEN TO BE + INFINITY.	