Abbreviations (cont.)

parset-n	parameter set for a procedure
text	any text
exp-list	expression list
var-list	list of variables
Basic-KSAM	Basic Keyed Sequential Access Method
fn	KSAM file number
pfn	primary file number
afn	alternate file number
prl	primary record length
pkey	primary key
akey	alternate key
pkl	primary key length
akl	alternate key length
akd	alternate key displacement
spc	unused space per block
...	continued as needed
| |	optional parameters
{ }	choose one of the enclosed parameters

Alphabetic List of Instructions

A

ABS
ADR
ASC
ATN
ATRIB
ATTR
AUTOL

B

BEGINCOMMON
BINADD
BINAND
BINOR
BINSUB
BINXOR
BYE

C

CALL
CHANGE
CHR$
CLEAR
CLOSE
COMMON
CON
COS
CREATE

D

DATA
DATE$
DEF
DEG
DELETE
DELREM
DIM
DIR
DO
DSK

E

ECHO
EDIT
ELSE
END
ENDCOMMON
ENDDO
ENDPROC
ENDWHILE
ENTER
ERRPROC
ERASE
ESC
EXITPROC
EXP
EXPAND

F

FIND
FRA
FOR
FRE

G

GET
GOSUB
GOTO

H

HEX$

I

IF
IMODE
INP
INPUT
INT
INTEGER
IOSTAT
IRN

K

KADD
KADDVOL
KALTADD
KALTCREATE
KALTCUR
KALTDEL
KALTFIRST
KALTFWD
KALTOPEN
KALTVER
KCLOSE
KCREATE
KDEL
KGET
KGETAPP
KGETBACK
KGETCUR
KGETFWD
KGETKEY
KGETREC
KLOAD
KOPEN
KPUT
KRETRIEVE
KUPDATE

L

LEN
LET
LFMODE
LIBRARY
LIST
LOAD
LOCAL
LOCK
LOG
LONG
LVAR

M

MAT
MAX
MIN

N

NOECHO
NOESC
NOLIST
NTRACE

O

ON ERROR
ON ESC
ON — GOTO
ON — GOSUB
OPEN
OUT

P

PEEK
POKE
POS
PRINT
PRINT USING
PROCEDURE
PUT

R

RAD
RANDOMIZE
READ
REM
REN
RENAME
RENUMBER
REPEAT
RESTORE
RETRY
RETURN
RND
RUN

S

SAVE
SCR
SET
SFMODE
SGN
SHORT
SIN
SPC
SQR
STOP
STR$
SYS

T

TAB
TAN
TIME$
TRACE

U

UNLOCK
UNTIL
USE
USR

V

VAL
VALC

W

WHILE

CROMEMCO 32K STRUCTURED BASIC

Instruction SYNTAX

YOUR LOCAL DEALER IS:

Cromemco 32K Structured Basic

Features

Control Structures facilitate modular programming.

Long Variable Names make program debugging and maintenance easier.

Statement Labels aid documentation and program comprehension.

In Line Basic Editor facilitates program changes.

Basic-KSAM allows data files to be accessed and records to be retrieved by specifying the contents of a key field.

Procedures allow for modular programming.

LVAR lists variables and current values.

DELREM deletes remark statements.

BEGINCOMMON & ENDCOMMON define a common storage area.

EXPAND inserts null characters in a string.

NOLIST generates run-only code.

HEX returns the ASCII hexadecimal representation of a number.

VALC performs error checking on user input.

TYPE returns the type of a numeric variable.

Cromemco™
incorporated
Tomorrow's Computers Today
280 BERNARDO AVE. MOUNTAIN VIEW, CA 94043

Part No. 023-9008 May 1980

Basic-KSAM

FILE HANDLING

KCREATE \prl,pkl[,spc]\ file-ref...	create primary data file
KCLOSE \fn\	close Basic-KSAM file
KOPEN \pfn\ file-ref...	open primary data file
KADDVOL \fn\ file-ref	add volume to existing file

SEQUENTIAL ACCESS

KGETBACK \pfn\ [var-list]	read previous record, primary file
KGETCUR \pfn\ [var-list]	read current record, primary file
KGETFWD \pfn\ [var-list]	read next record, primary file
KGET \pfn\ var-list	read from current record, primary file
KPUT \pfn\ var-list	write to current record, primary file
KRETRIEVE \pfn\svar	retrieve primary key, current record

RANDOM ACCESS

KGETKEY \pfn,pkey\ [var- list]	read random record, primary file
KGETAPP \pfn,pkey\ [var- list]	read approximate, primary file
KUPDATE \pfn,pkey\ [var- list]	update record, primary file
KDEL \pfn,pkey\	delete record, primary file
KGETREC \pfn,rec-num\ [var-list]	read Nth record, primary file
KADD \pfn,pkey\ [var-list]	add record, primary file
KLOAD \pfn,pkey\ [var- list]	load record, primary file

ALTERNATE KEY ACCESS

KALTCREATE \pfn,akl [,akd]\ file-ref...	create alternate key file
KALTOPEN \afn,pfn\ file-ref...	open alternate key file
KALTCUR\afn\ [var-list]	read primary record by current alternate key
KALTFIRST \afn,akey\ [var-list]	read next primary record by specified alternate key
KALTFWD \afn\ [var-list]	read next primary record by current alternate key
KALTVER \afn\	verify alternate record
KALTADD \afn\	add record, alternate file
KALTDEL \afn\	delete record, alternate file

Abbreviations

N-n	line number
L-n	line name or number
c	command
f	function
s	statement
file-ref	file reference
chnl	file channel number
mode	file access mode
byte-num	byte number
rec-num	record number
rec-size	record size
avar	arithmetic variable
mvar	matrix (dimensioned) variable
svar	string variable
var	any variable
aexp	arithmetic expression
exp	expression
sexp	string expression
dum	dummy parameter
.prname	procedure name

Program Development

c AUTOL N-1, N-2	automatic line numbering
{ATTR ATRIB} file-ref ,svar	alter file attributes
BYE	exit from Basic
DELETE [{L-1 L-1, L-1,L-2}]	delete statement lines
c DIR [file-ref]	directory
ENTER file-ref	enter ascii file
LIST [file-ref,] [{L-1 L-1, L-1,L-2}]	list current program
LVAR [file-ref]	list variables
LOAD file-ref	load binary file
c RE-NUMBER [{N-1 N-1,N-2 N-1,N-2,L-3 N-1,N-2,L-3, N-1,N-2,L-3,L-4}]	renumber statement lines
RUN [file-ref]	execute program
SAVE file-ref	save current program
SCR	scratch user area
TRACE	enable trace option
NTRACE	disable trace option

In Line Editor

c EDIT [{L-1 L-1, L-1,L-2}]	edit program
D	delete character
I	insert text
K	delete the rest of line
c FIND [{L-1 L-1, L-1,L-2}]	find string
c CHANGE [{L-1 L-1, L-1,L-2}]	change string
carriage return	reject change
C	accept change
`*`	change all following occurrences

Documentation

REM text	remark

Assignment Operator

LET var = exp	assignment
MAT mvar = aexp	matrix initialization

Arithmetic Operators

(in order of precedence)

+	unary plus
−	unary minus
`**` or ^	exponentiation
`**`	multiplication
/	division
+	addition
−	subtraction

Relational Operators

=	equal to
<	less than
>	greater than
<=	less than or equal to
>=	greater than or equal to
<> or #	not equal to

Initialization

DEG	degree mode
DIM {svar (aexp-1) avar (aexp-1[,aexp-2 [,aexp-3]])}	dimension variable
IMODE	integer mode
INTEGER {avar mvar}	integer variable
LFMODE	long floating point mode
LONG {avar mvar}	long floating point variable
RAD	radian mode
SFMODE	short floating point mode
SHORT {avar mvar}	short floating point variable

Control Structures

c CON	continue program execution
s END	halt program execution
s FOR avar = aexp-1 TO aexp-2 [STEP aexp-3] s NEXT avar	for-next loop
GOSUB L-1	subroutine call
{RETURN	standard return
RETRY}	optional return
GOTO L-1	transfer control
IF aexp THEN instruction: ...	if-then conditional branch
s IF aexp THEN DO s [ELSE] s ENDDO	if-then-else conditional branch
ON aexp {GOTO GOSUB} L-1,L-2,...	multi-path branch
s REPEAT s UNTIL aexp	repeat-until loop
s WHILE aexp s ENDWHILE	while-endwhile loop
s STOP	stop program execution
s DO s ENDDO	define logical program segment

Console and Data I/O

INPUT ["text",] var-list	input from console
PRINT [exp-list]	print to console
SPC(aexp)	space function
TAB(aexp)	tab function
PRINT USING svar, exp-list	print using, svar may contain: # leading blanks & leading zeroes `*` leading asterisks , comma . decimal point + fixed or floating plus sign − fixed or floating minus sign $ fixed or floating dollar sign !!!! exponent field
READ var-list	read data statements
RESTORE [L-1]	restore data pointer
s DATA exp-list	data statement

File I/O

CREATE file-ref	create file
OPEN \file-num[,rec-size [,mode]]\ file-ref	open file for processing
CLOSE [\file-num\]	close file
ERASE file-ref	erase file
RENAME old-file-ref, new-file-ref	rename file
REN new-file-ref,old-file-ref	rename file
PRINT \file-num[,rec-num[,byte-num]]\ [exp-list]	print to a file (ascii)
INPUT \file-num[,rec-num [,byte-num]]\ [var-list]	input from a file (ascii)
PUT \file-num[,rec-num [,byte-num]]\ [exp-list]	put to a file
GET \file-num[,rec-num [,byte-num]]\ [var-list]	get from a file

Note: All file I/O optional parameter definitions pertain to disk files only.

Functions

f ABS(aexp)	absolute value
f BINADD(aexp-1,aexp-2)	binary addition
f BINAND(aexp-1,aexp-2)	binary logical And
f BINOR(aexp-1, aexp-2)	binary logical Or
f BINSUB(aexp-1,aexp-2)	binary subtraction
f BINXOR(aexp-1,aexp-2)	binary logical Exclusive Or
f EXP(aexp)	"e" to the power X
f FRA(aexp)	fractional portion
f INT(aexp)	integer value
f IRN(aexp)	integer random number generator
f LOG(aexp)	natural logarithm
f MAX(aexp-1,...,aexp-n)	maximum value
f MIN(aexp-1,...,aexp-n)	minimum value
RANDOMIZE	used with Rnd and Irn
f RND(aexp)	random number generator
f SGN(aexp)	algebraic sign
f SQR(aexp)	square root

Trigonometric Functions

f ATN(aexp)	arctangent
f COS(aexp)	cosine
f SIN(aexp)	sine
f TAN(aexp)	tangent

Programmer Defined Functions

f DEF FNname(X-1,...,X-n) = aexp	user defined function

String Functions

f ASC(sexp)	ASCII value
f CHR$(aexp)	ASCII equivalent
EXPAND svar, aexp	inserts null characters
f HEX$(aexp)	ASCII hexadecimal representation
f LEN(sexp)	length of string.
f POS(sexp-1,sexp-2, avar)	position of substring
f STR$(aexp)	character representation
f VAL(sexp)	numeric representation
f VALC(sexp)	numeric representation with error conditions
f DATE$(sexp)	date
f TIME$(sexp)	time

System and File Status

DSK [svar]	display or alter current disk drive, eject disks, turn drive motors off
ECHO	re-enables display of user input
NOECHO	disables display of user input
ESC	re-enables escape key operation
NOESC	disables escape key operation
f FRE(dum)	free space
f IOSTAT(file-num,aexp-1)	I/O status
ON ERROR {STOP GOTO L-1 GOSUB L-1}	on error transfer control
ON ESC {STOP GOTO L-1 GOSUB L-1}	on escape transfer control
SET aexp-1,aexp-2	set system parameter
f SYS(aexp)	system information
f ADR(var)	address of variable

Machine Level

f INP(aexp)	input from port
OUT aexp-1, aexp-2	output to port
f PEEK(aexp)	contents of memory location
POKE aexp-1, aexp-2	output to memory location
f USR(aexp-1,aexp-2,...)	call a user assembly language program

Scope

COMMON	reserve common storage area, method I
BEGINCOMMON	begin common storage area, method II
ENDCOMMON	end common storage area, method II
LOCAL {avar svar MAT mvar} ,....	define local variable

Procedures

[CALL] .prname [(parset-1; parset-2)]	procedure call
s PROCEDURE .prname [(parset-1)]	procedure definition
s ENDPROC [(parset-2)]	procedure end
s ERRPROC	procedure error end
s EXITPROC [(parset-2)]	procedure exit
CLEAR {aexp .prname}	clear partition
LIBRARY [file-ref]	open or close library
c USE {aexp .prname}	use partition
LOCK {aexp .prname}	lock partition
UNLOCK {aexp .prname}	unlock partition

Program Security

DELREM [{L-1 L-1, L-1,L-2}]	delete remarks
NOLIST [{L-1 L-1, L-1, L-2}]	disable listing