Business BASIC Level 3&4 Reference Card

VARIABLES

CTL	Field terminator last used
DAY	Current system date
DSZ	Unused bytes in user task memory
ERR	Last occurring error
HSA	Highest sector available
PSZ	Bytes used by program
SSN	System Serial Number
SSZ	Bytes in sector
SYS	Level of operating system (Level 4 only)
TCB	Task information
TIM	Current system time
TSK(0)	Configured devices
TSK(1-9)	Tasks in bank

INPUT/OUTPUT OPTIONS

BLK=	User buffer size (Level 3 only)
BNK=	Bank number
DOM=	Duplicate or missing key
END=	Branch at end of file
ERR=	Branch on error
IND=	Access record index
IOL=	Specifies IOLIST stno
ISZ=	Allows a file to be accessed like an Index file
KEY=	Key of record to be accessed
LEN=	Length range of variable
RTY=	Number of retries
SEQ=	File position on MTC or MTR
SIZ=	Maximum characters to be input
TBL=	Specifies TABLE stno
TIM=	Seconds allowed for input
TRK=	Track number of tape cartridge
VOL=	Volume (magnetic tape only) (early Level 3 only)

FUNCTIONS

ABS	Absolute value
AND	Logical ANDing
ASC	String to decimal
ATH	Hexadecimal value to ASCII
BIN	Binary value
BSZ	Bytes available in bank
CHR	Numeric expr to ASCII
CPL	Compilestring expr
CRC	Checksum for string variable; data integrity check
DEC	Binary to signed decimal
EPT	Exponent of expr
FID	File information
FNx	Definable functions
FPT	Fractional part of expr
GAP	Odd parity, byte-for-byte
HSH	HASH; data integrity check
HTA	ASCII value to hexadecimal
IND	Index of record
INT	Integer of expr
IOR	Inclusive ORing
KEY	Key of next record
LEN	Length of string expr
LRC	Longitudinal Redundancy check; data integrity check
LST	Compiled BASIC to List format
MOD	Remainder of divided integers
NOT	Inverse of string, bit-by-bit
NUM	String expr to numeric
PGM	Compiled format of stno
POS	Character position
PUB	Public Programs in bank
SGN	Sign of expr
STR	Numeric expr to string
XOR	Exclusive ORing

BFISD 5016

STATEMENT FORMATS

* = Level 3 only
** = Level 4 only

```
ADD "prog ID" {,ERR=}
ADDR "prog ID" {,ERR=} {,BNK=}
ADDE "OSSPOL" {,ERR=} {,BNK=}
ADDC* ".CPLR" {,ERR=} {,BNK=}
ADDL* ".LSTR" {,ERR=} {,BNK=}
ADDS** ".SORT" {,ERR=} {,BNK=}

BEGIN

CALL "prog ID" {,ERR=} {,SIZ=}** {,argument list}
CLEAR
CLOSE (fileno/devno {,ERR=} {,IND=})

DEF FNx{$} (argument list)=expr
DELETE {first stno} {,} {last stno}


DIM array name (range of first dimension {,range of second {,range of third }})
DIM string variable name (length {,str expr})
DIRECT "file ID", keysz, recno, recsz, discno, secno {,ERR=}
DISABLE discno
DROP "prog ID" {,ERR=}

EDIT stno {C[copy through value]} {D[delete through value]} {R[replace value]} {[insert value]}
ENABLE discno
END
ENDTRACE
ENTER argument list
ERASE "file ID" {,ERR=}
ESCAPE
EXECUTE {stno} string argument
EXIT {expr}
EXITTO stno
EXTRACT (fileno {,ERR=} {,END=} {,DOM=} {, IND=} {,KEY=} {,TBL=} {,SIZ=}) {argument list}{ IOL=}
EXTRACT RECORD (fileno {,ERR=} {,END=} {,DOM=} {,IND=} {,KEY=} {,TBL=} {,SIZ=}) {string variable }
```

```
FILE string
FIND (fileno {,ERR=} {,END=} {,DOM=} {,KEY=} {,TBL=} {,SIZ=}) {argument list} {,IOL=}
FIND RECORD (fileno {,ERR=} {,END=} {,DOM=} {,KEY=} {,TBL=} {,SIZ=}) {argument list}
FLOATING POINT
FOR crtl variable = start expr TO end expr  STEP { expr}

GET discno, secno {,ERR=} {,RTY=}, input string variable {,verify string variable}
GOSUB stno
GOTO stno

IF logical expr {AND logical expr} {OR logical expr} {THEN} statement {ELSE statement}
INDEXED "file ID", recno, recsz, discno, secno {,ERR=}
INPUT {(fileno/devno {,ERR=} {,END=} {,DOM} {,IND=} {,KEY=} {,TBL=} {,TIM=} {,SIZ=})} { @(expr{,expr})} {,string constant} {,mnemonic}
   {,variable} {,IOL=}
INPUT RECORD (fileno/devno {,ERR=} {,END=} {,DOM=} {,IND=} {,KEY=} {,SIZ=}) {@(expr{,expr})} {string variable}
IOLIST argument list {,IOL=}

{LET} {numeric variable=numeric expr} {,} {string variable=string expr} {,...}
LIST {(devno {,ERR=} {,TBL=})} {first stno} {,} {last stno}
LOAD "prog ID"
LOCK (fileno {,ERR=})

MERGE (fileno/devno {,ERR=} {,IND=} {,TBL=})


NEXT control variable

ON expr GOTO stno {,stno, stno...}
OPEN(fileno/devno {,ERR=} {,BLK=} {,TRK=} {,SEQ=} {,ISZ=}) "file/device ID"

PRECISION expr
PRINT {(fileno/devno {,ERR=} {,END=} {,IND=} {,KEY=} {,DOM=} {,TBL=})} {@(expr {,expr})} {list} {,IOL=} {,}
PRINT RECORD (fileno/devno {,ERR=} {,END=} {,SIZ=} {,DOM=} {,IND=} {,KEY=} {,TBL=}) {@(expr {,expr})} {string variable}
PROGRAM "file ID", prog size, discno, secno {,ERR=}
PUT discno, secno {,ERR=} {,RTY=}, string variable {,verify string variable}

READ {(fileno/devno {,ERR=} {,END=} {,IND=} {,KEY=} {,TBL=} {,SIZ=} {,DOM=} {,TIM=})} {@(expr {,expr})} {variable list} {,IOL=}
READ RECORD (fileno/devno {,ERR=} {,END=} {,IND=} {,KEY=} {,TBL=} {,DOM=} {,TIM=} {,SIZ=}) string variable
RELEASE {"task ID"}
REM {"} {remark} {"}
REMOVE (fileno, KEY= {,DOM=} {,ERR=})
RESERVE discno
RESET
RETRY
RETURN
RUN {"prog ID"}
```

SAVE "prog ID" {,prog size, discno, secno}
SERIAL "file ID", average recno, average recsz, discno, secno {,ERR=}
SETCTL** stno
SETDAY "string expr"
SETERR stno
SETESC stno
SETTIME numeric expr
SETTRACE {(fileno/devno)}
SORT "file ID", keysz, recno, discno, secno {,ERR=}
SORTSTEP** string expr, string expr, string expr
START pages {,ERR=} {,BNK=} {,"prog ID"} {,"task ID"}
STOP

TABLE hexadecimal string

UNLOCK (fileno/devno {,ERR=})

WAIT number of seconds
WRITE {(fileno/devno {,ERR=} {,END=} {,DOM=} {,IND=} {,KEY=} {,SIZ=} {,TBL=})} {@(expr {,expr})} {variable list} {,IOL=}
WRITE RECORD (fileno/devno {,ERR=} {,END=} {,DOM=} {,IND=} {,KEY=} {,SIZ=} {,TBL=}) {string variable}

ERRORS

ERROR TYPE	ERROR CODE	MEANING
INPUT/OUTPUT	00	FILE/RECORD/DEVICE BUSY OR INACCESSIBLE
	01	END OF RECORD
	02	END OF FILE
	03	DISC READ ERROR
	04	DISC NOT READY
	05	PERIPHERAL DATA TRANSFER ERROR
	06	INVALID DISC DIRECTORY OR NON-CERTIFIED TAPE CARTRIDGE
	07	SECTOR/POINTER OUT OF RANGE
	08	DISC WRITE ERROR/ DATA MISCOMPARE
	09	POWER FAILURE
FILE USAGE	10	ILLEGAL FILE NAME SIZE OR USAGE/ILLEGAL OVERLAID CALL
	11	MISSING OR DUPLICATE KEY
	12	MISSING OR DUPLICATE FILE NAME/NON-CONFIGURED DEVICE
	13	IMPROPER FILE OR DEVICE ACCESS
	14	IMPROPER FILE OR DEVICE USAGE
	15	DISC SPACE OCCUPIED BY ANOTHER FILE
	16	DISC OF PUBLIC PROGRAMMING DIRECTORY IS FULL
	17	INVALID PARAMETER/ NON-CONFIGURED DISC
	18	ILLEGAL CONTROL OPERATION
	19	PROGRAM SIZE IS LARGER THAN DEFINED SIZE
STRUCTURE	20	STATEMENT SYNTAX
	21	INVALID STATEMENT NUMBER
	23	MISSING VARIABLE/ NON-DIMENSIONED STRING
	24	DUPLICATE FUNCTION NAME
	25	UNDEFINED FUNCTION
	26	INCORRECT VARIABLE USAGE
	27	RETURN WITHOUT GOSUB/DELETE WITH ACTIVE GOSUB OR FOR-NEXT
	28	NEXT WITHOUT FOR
	29	UNDEFINED MNEMONIC
LIMIT	30	USER PROGRAM INCORRECT CHECKSUM
	31	INSUFFICIENT MEMORY WITHIN TASK
	32	(HARDWARE) STACK OVERFLOW
	33	INSUFFICIENT MEMORY CAPACITY
	34	VDT BUFFER OVERFLOW
	35	PARENTHETIC EXPRESSION LIMIT
	36	CALL/ENTER VARIABLE MISMATCH
	37	INVALID FUNCTION
	38	ILLEGAL COMMAND IN PUBLIC PROGRAM
	39	ESCAPE IN PUBLIC PROGRAM
EXECUTION	40	NUMERIC VALUE OVERFLOW
	41	INVALID INTEGER RANGE
	42	NON-EXISTENT NUMERIC SUBSCRIPT
	43	INVALID FORMAT MASK SIZE
	44	STEP SIZE OF ZERO
	45	INVALID STATEMENT USAGE
	46	INVALID STRING SIZE
	47	SUBSTRING REFERENCE OUT OF RANGE
	48	INVALID INPUT
	49	NON-TRANSLATABLE STATEMENT
	50	GENERAL MEMORY ERROR
	51 (L3)	COMPILE OR LIST OPERATION WITHOUT COMPILER/LISTER
	51 (L4)	SORTSTEP NOT RESIDENT
	54	OPEN OF SERIAL FILE WITH INVALID HEADER
	55	TAPE CONTROLLER
COMMUNICATIONS	72	END OF TRACK ON TAPE/ UNEXPECTED ETX
	75	NEGATIVE BID RESPONSE
	77	REVERSE INTERRUPT

ERROR TYPE	ERROR CODE	MEANING
	80	INVALID EXPECTED BID
	81	BID RECEIVE TIMEOUT
	82	BID RESPONSE RETRY LIMIT
	83	INVALID ID SEQUENCE
	84	PRIMARY/SECONDARY ASSIGNMENT
	85	DATA TIMEOUT
	87	RESPONSE-TO-DATA, RETRY LIMIT
	88	RETRANSMISSION LIMIT
	89	ACKNOWLEDGEMENT PHASE
	95	DEVICE ERROR
CATASTROPHIC INPUT/OUTPUT	103	CATASTROPHIC READ FAILURE/FILE POINTERS DAMAGED
	104	CATASTROPHIC DISC FAILURE/FILE POINTERS DAMAGED
	108	CATASTROPHIC WRITE FAILURE
	123	CATASTROPHIC PARITY ERROR/FILE POINTERS DAMAGED
	124	PARITY ERROR
SPECIAL CODE	126	(L4)CTL+Y USED
ESCAPE	127	ESCAPE

MNEMONICS

MNEMONIC	APPLICABLE LEVEL	FUNCTION
@(x)	3,4	Horiz position
@(x,y)	3,4	Horiz & vert position
'BE'	4	Begin echo
'BG'	4	Begin to generate ERROR 29
'BI'	4	Begin input transparency
'BO'	4	Begin output transparency
'BS'	3,4	Backspace
'BT'	4	Begin input buffering
'CE'	4	Clear screen to end of page
'CF'	3,4	Clear foreground
'CH'	3,4	Move cursor home (0,0)
'CI'	3,4	Clear input buffer
'CL'	3,4	Clear line
'CR'	3,4	Carriage Return
'CS'	3,4	Clear screen
'DC'	4	Delete character
'EE'	4	End echo
'EG'	4	End generation of ERROR 29
'EI'	4	End input transparency
'EL'	3,4	End Load (VFU)
'EO'	4	End output transparency
'EP'	3,4	Expanded print
'ES'	3,4	Escape
'ET'	4	End input buffering
'FF'	3,4	Form feed
'IC'	4	Insert character
'LD'	3,4	Line delete
'LF'	3,4	Line feed
'LI'	3,4	Line insert
'PE'	4	End protect mode
'PG'	3,4	Transmit screen to local serial printer
'PM'	3,4	Begin plot mode
'PS'	4	Start protect mode
'RB'	3,4	Ring bell
'RC'	4	Read cursor position
'SB'	3,4	Start background mode
'SF'	3,4	Start foreground mode
'SL'	3,4	Start load (VFU)
'S2'	3,4	Slew to channel 2
'S3'	3,4	Slew to channel 3
'S4'	3,4	Slew to channel 4
'S5'	3,4	Slew to channel 5
'S7'	3,4	Slew to channel 7
'S8'	3,4	Slew to channel 8
'TR'	3,4	Transmit VDT screen
'VT'	3,4	Vertical tab

RELATIONAL OPERATORS

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

LOGICAL/ARITHMETICAL OPERATORS

- − minus
- + plus
- * multiplied by
- / divided by
- ^ raised to the power of (exponentiation)