TERSE Standard Glossary

This is a description of the vocabularies. The words are presented in ASCII order. The first line of each entry shows a symbolic description of the action of the word: Symbols indicating which parameters are to be placed on the stack before executing the word, 3 dashes (---) indicating execution, then any parameters left on the stack by the word. In this notation, the top of the stack is to the right. If the place of the word in the input string is not completely obvious, it is shown explicitly. If no dashes are shown the word does not affect the stack. Symbols are used as follows:

b	Block number
c	7-bit ASCII character code
f	Flag: 0=False, non-zero=True. All words which return a flag return 0 or 1.
m n p q r	16-bit integers
nnnn pppp	The name of a word
ssss	A string of characters

Immediately following the name of a word, certain characters may appear within paraentheses. These denote some special action or characteristics:

C	The word may be used only within a colon-definition. A following digit (C0 or C2) indicates the number of memory cells used when the word is compiled if other than one. A following + or - sign indicates that the word either pushes or pops a value on the stack during compilation. (This action is not related to its action during execution.)
E	The word may not normally be compiled within a colon-definition.
P	The word has its immediate bit set; it is executed directly, even when encountered during compile mode.
U	The word applies to a user variable (in a multi-user system each user would have his own copy.)

Unless stared otherwise, all references to numbers apply 16-bit integers, with the most significant bit as the sign bit and the negative in two's complement form. Similarly, all arithmetic will be assumed to be 16-bit signed integer arithmetic with error and overflow indication unspecified.

Standard Definitions

! m p ---

Store m at address p.

' (P) ' nnnn --- p

Leave address of verb nnnn on stack. A compiler directive, ' is executed when encountered in a colon definition; the address of the following word's code field is found immediately (at compilation) and stored in the dictionary (after the address of LIT) as a literal to be placed on the stack at execution time. Within a colon definition, ' nnnn is identical to: LIT [' nnnn ,]

((P) (ssss)

Ingore a comment that will be delimited by a right parenthesis.

***** m n --- p

16-bit signed multiply. p=m*n

+ m n --- q

16-bit integer addition. q=m+n

+! m p ---

Add integer m to value at address p.

+BLOCK m --- b

Leave the sum of m plus the number of the block currently being interpreted.

+LOOP (C) m ---

Add m to the loop index. exit from the loop is made when the resulant index reaches or passes the limit, if m is greater than zero; or when the index is less than (passes) the limit, if m is less than zero. m may be variable.

, m ---

Store m into the next available dicitionary cell, advancing the dictionary pointer.

- m n --- q

16-bit integer subtraction: q=m-n

--> (P)

(Pronounced "next block") Continue interpretation with the next block (Equivalent to 1 +BLOCK CONTINUED).

. m ---

Print the value on the stack as an integer, converted according to the current number base.

." (P) ." ssss"

Transmit a message delimited by " to the selected output device.

.LIST

Change output device to CRT.

.NLIST
Change output device to PRINTER.

/ m n --- q
16-bit signed integer divide, q=m/n. The quotient is truncated and the remainder is lost. (Actually defined as /MOD DROP)

/MOD m n --- r q
16-bit integer divide, m/n. The quotient is left on top of the stack, the remainder beneath. The remainder has the sign of the quotient, q.

0 --- n
Puts a 0 on the stack. (0 is a constant)

0< m --- f
Leave a true flag if m is negative.

0= m --- f
True if m is zero.

0> m --- f
True if m is positive and non-zero.

0END BEGIN.....0END
Mark the end of a BEGIN loop. Causes endless loop.

1 --- n
Puts a 1 on the stack. (1 is a CONSTANT)

1+ m --- q
q=m+1

1+! p ---
Add 1 to the contents of location p

1- m --- q
q=m-1

1-! p ---
Subtract 1 from the contents of location p.

2* m --- q
q=m*2 (Shift left)

2+ m --- q
q=m+2 (Increment by 2)

2- m --- q
q=m-2 (Decrement by 2)

2/ m --- q
q=m/2 (Shift Right)

2DROP m n ---
Drop the top two values from the stack (to drop a double precision number for example).

```
2DUP                    m n --- m n m n
         Duplicate the top two values on the stack.

2SWAP                   m n p q --- p q m n
         Swap two pairs of values (e.g. double-precision numbers).

:                       : nnnn
         Create a dictionary entry defining  nnnn as equivalent to the
         following sequence of TERSE words. Set STATE to compile mode.
         (Extension: Set  the  context  vocabulary  equivalent  to the
         current vocabulary).

;        (C,P)
         Terminate a colon-definition and set STATE to immediate mode.

;S       (E)
         Stop interpretation of a symbolic block.

<                       m n --- f
         True if m<n

<<       (E)            m n ---
         Use: m n << verbs >>
         Unlike DO...LOOP  these conditionals  may be  employed during
         interpretation.  In conjuction with  the words  [ and  ] they
         may be used within a colon definition to control compilation,
         although they are not compiled.  These words can be nested.

<=                      m n --- f
         True if m<n or m=n

<>                      m n --- f
         True if m is unequal to n

=                       m n --- f
         True if m=n

>                       m n --- f
         True if m>n

>=                      m n --- f
         True if m>n or m=n

>>       (E)
         Terminate a conditional interpretation sequence begun by <<.

>R  (C)                 m ---
         Push m onto top of the return stack. ( See R> )

?                       p ---
         Print the value contained at address according to the current
         base.

@                       p --- q
         Leave contents q of memory location p.

A"               A" ssss" --- q
         Makes a string similarly to ."  but does not type it. Instead
```

A" returns the string address q. The string may be typed by COUNT TYPE or STYPE.

ABORT

Enter the abort sequence, reset stack and return stack, return control to terminal, and print an abort message ("?" beep).

ABS m --- q

Leave the absolute value of a number.

AND m n --- q

Bitwise logical AND of m and n.

ARRAY m ARRAY nnnn ---

Define an array named nnnn and allocate m uninitialized words into the dictionary (or RAM). The sequence i nnnn will leave the address of the i-th cell on the stack. The index should be in the range 0 <= i <= m-1, but no check is made for values exceeding this range.

ASM (P)

Switch the context pointer so that dictionary searches will begin at the Assembler Vocabulary. A CODE define automatically switches the CONTEXT to ASM.

B! m p ---

Store the least significant 8 bits of m at byte-address p.

B: m --- q

q=m+308. Used for calculating block numbers of drive B.

B@ p --- q

Return the 8-bit byte q found at byte-address p.

BARRAY m ARRAY nnnn ---

Define an array named nnnn and allocate m uninitialized bytes into the dictionary (or RAM). The sequence i nnnn will leave the address of the i-th byte on the stack. The index should be in the range 0 <= i <= m-1, but no check is made for values exceeding this range.

BASE (U) --- p

A variable containing the current number conversion base.

BEGIN (C0+,P) BEGIN ... WHILE ... REPEAT
or BEGIN ... END

Mark the start of a sequence of words to be executed repetitively. If ... WHILE ... REPEAT is used the loop will be repeated as long as the stack encountered by WHILE is TRUE (REPEAT merely effects an unconditional jump back to BEGIN); when WHILE sees a FALSE value (0) on the stack it causes an immediate exit out of the loop. In case the sequence can be written such that the test for completion is at the end ... END can be used conveniently to end the loop on a TRUE value or to go back to BEGIN on FALSE. Both WHILE and END drop the value they test.

BELL

Sends a BELL char to the terminal

BLK (U) --- p

A variable containing the number of the block being listed or edited.

BLOCK b --- p

Leave the first address of Block b. If the block is not already in memory, it is transferred from disk into whichever core buffer has been least recently accessed. If the block occupying that buffer has been updated, it is rewritten on disk or tape before Block b is read into the buffer.

BMOVE $\overset{\curvearrowright}{p\ q}$ n ---

Move the n bytes starting at byte-address p into the n byte-cells starting at byte-address q. The contents of p is moved first.

BPTR --- n

A variable containing a pointer to the most recently used disk block buffer. Disk block buffers are headed by a link to the next block and the block number followed by the data. A link of 0 indicates the end of the chain.

BTA n ---

Convert the value n to a character string at the next available dictionary location (HERE) leaving the character count in the first byte. Leading spaces are added to make the total number of characters equal the value of the variable FLD.

BTABLE BTABLE nnnn ----

Define the beginning of a table of bytes. The values to be entered into the table must follow the definitions of the table. The sequence i nnnn will leave the address of the i-th byte on the stack. The index should be 0 <= i < number-of-table-entries. No check is made on the range of i.

BUFFER b --- p

Obtain a core buffer for Block b, leaving the first buffer cell address. The block is not read from disk, and is automatically marked as updated.

BUILD BUILD pppp ----

Read the next word from the imput stream, see if it already exists in the CONTEXT vocabulary (if so, print the warning, "pppp already defined"), then enter it into the dictionary in the CURRENT vocabulary.

BYE

Exit to ICEbox monitor.

BYTE p --- q

Return the byte address of the first byte in memory cell p.

```
CASE      (C2+,P)   m n --- (m)
                    m n CASE <action for m=n> ELSE <drop> THEN
          If m equals n,  drop  both  m  and  n  and  execute the words
          directly following  CASE  until  the  next  ELSE  or  THEN  ;
          otherwise, drop n  but leave  m and  execute the  words after
          ELSE (or THEN if  no ELSE is used).  The  selection of one of
          many cases can be done by:
                    m n1 CASE <action for m=n1> ELSE
                      n2 CASE <action for m=n2> ELSE
                      n3 CASE <action for m=n2> ELSE
                         <otherwise action> THEN THEN THEN
          (m will still be on the stack in the otherwise section).

CCALC               m --- q
          Converts a link address  m  to  the  code  address  q of that
          routine.

CIN                 --- n
          Leaves the address n of a CRT input routine on the stack.

CODE                CODE nnnn
          Create a dictionary entry defining  nnnn as equivalent to the
          following sequence of  assembler  code.  (Extension:  set the
          context vocabulary to Assembler.)

COM                 m --- q
          Complement each bit of m (Leave one's complement).

CONSTANT            m CONSTANT nnnn ---
          Create a word which  when executed  pushes m  onto the stack.
          Since the "constant"  m maybe  modified by  the sequence  q '
          nnnn 3 + ! it is oftentimes advantageous to define a variable
          as a constant, particulary if it  is accessed more than it is
          modified.

CONTEXT   (U)       --- p
          A variable containing  a pointer  to the  vocabulary in which
          dictionary searches are to begin. See CURRENT.

CONTINUED (E)       b ---
          Continue  interpretation  at  block  b.  (The  preferred
          implementation in multi-buffer systems is such that the block
          buffer currently being accessed will  be used for storeage of
          block b, leaving other buffers unaffected.)


COPY                m n ---
          Copies block m to block n.

COUNT               p --- m n
          Leave byte-address  m and  byte-count n  of a  message string
          beginning at word-address  p.  It is presumed  that the first
          byte at p contains the byte-count and that the actual message
          starts with the second byte  in location p.  Typically, COUNT
          is followed by WRITE or TYPE

COUT                --- n
          Leaves the address n of a CRT output routine on the stack.
```

CR

Transmit carriage return/linefeed codes to the selected output device.

CURRENT (U)

A variable containing a pointer to the vocabulary into which new words are to be entered. CURRENT @ @ leaves the link address of the next entry to be defined.

DECIMAL

Set the numeric conversion base to decimal mode.

DELIM --- q

A variable containing the ASCII character used as a delimiter by WORD.

DGTS --- q

A variable containing the number of digits to the right of the decimal point in the most recently converted number. If there was no decimal point then it is the number of digits.

DIR m n ---

Lists the first line of each block that starts with "(" from block n to block m-1.

DISKCOPY

Copys all blocks from disk drive A to drive B.

DLIT (C) DLIT l h

Automatically compiled before each double precision literal encountered in a colon definition. Execution of DLIT causes the contents of the next 2 instruction words to be pushed onto the stack. High value is on top.

DO (C) m n ---

Begin a loop, to be terminated by LOOP or +LOOP. The loop index begins at n, and may be modified at the end of the loop by any positive or negative value. The loop is terminated when an increment index reaches or exceeds m, of when a decremented index becomes less than m.
Within nested loops, the word I always returns the index of the innermost loop that is being executed, while J returns the index of the next outer loop, and K returns the index of the second outer loop.

DP --- q

A variable containing a pointer to the next available dictionary location.

DP+! n ---

Add the signed value n to the dictionary pointer (DP). As DP may be an internal register rather than a VARIABLE, it is accessible only through HERE and DP+!

DPREC --- q

A variable containing a flag indicating if the most recently converted number was double precision. (TRUE=Double Precision). The following characters cause double precision conversion: . / , - (as a dash)

DROP m ---
Drop the top value from the stack.

DUMP m n ---
Dump the contents of n memory cells starting at address m. Normally, both addresses and contents are shown in the current base.

DUP m --- m m
Duplicate the top value on the stack.

E.B --- q
A variable used by ABORT. When an abort is made the current block number is stored into E.B. Block 0 is the keyboard input buffer.

E.O --- q
A variable used by ABORT. When an abort is made the current offset in the current block (IOFF) is stored into E.O.

EDIT (P)
The name of the Edit Vocabulary. If that vocabulary is loaded, EDIT establishes it as the context vocabulary, thereby making its definitions accessible.

ELSE (C2,P)
Precede the false part of an IF...ELSE...THEN conditional. It may be ommitted if the false part is empty.

EMPTY --- q
A variable containing a flag indicating if a CR has been scanned by WORD. (TRUE=CR scanned)

END (C2-,P) f ---
Mark the end of a BEGIN..END loop. If f is true the loop is terminated. If f is false, control returns to the first word after the corresponding BEGIN.

ENTER
Creates a dictionary entry using the packed character string at the HERE with character count in first byte. HERE is left pointing to the parameter field.

ERASE-CORE
Marks all block-buffers as empty. Updated blocks are not flushed. Contents of buffers are undefined.

EXEC q ---
Depending on the STATE variable either q is stored in the dictionary or address q is loaded into HL and a PCHL is done.

EXPECT n ---
Gets ASCII character input from the selected input device. Input is terminated with the first carraige return or after n characters have been accepted.

FILECOPY m n ---
Copies blocks n thru m from drive A to drive B.

FIND --- p
Returns the address of nnnn (i.e. the address of the link field) if nnnn can be found in the dictionary; otherwise skips two words in the definition.

FLD --- p
A variable containing the field length reserved for a number during output conversion.

FLUSH
Write all blocks that have benn flagged as "updated" to disk. Return when output is done.

FNAME FNAME nnnn ---- m p
Find name nnnn in CONTEXT vocabulary (search dictionary). If found, return address of link as m and p set to 1. If not found, m is omitted and p is 0.

GETC --- n
Inputs an ASCII character n from the selected input device.

GOODBYE
Writes out updated disk-buffers and exits to monitor.

H. m ---
Convert and output in hexadecimal mode, unsigned, and preceded by a blank. BASE is unchanged. Format specifications are observed.

HELP (E) ---
List the dictionary, starting LAST @ . This starts with the CONTEXT vocabulary.

HERE (U) --- p
Return the address of the next available dictionary location.

HEX
Switch the numeric conversion base to hexadecimal.

HEXLIST m b ---
List the ASCII contents and hexadecimal contents of block b starting at byte m on the selected output device.

HEXSHOW b ---
Lists ASCII contents and hexadecimal contents of block b on the selected output device. Repeated pressings of the space bar on the control terminal will list the next 256 bytes of the block. Pressing any other key will terminate the sequence.

I (C) --- m
Returns the index of an intermost DO-loop.

I+ m --- q
Adds m to the index of the intermost DO-loop. q=m+I

```
IF          (C2+,P)   f IF <true part> ELSE <false part> THEN
                      f IF <true part> THEN
            IF s  the first  word of  a conditional.  If  f is  true, the
            words following IF are executed  and the words following ELSE
            are not  executed.  The  ELSE  part  of  the  conditional  is
            optional.  If f is  false,  words  between  IF  and  ELSE, of
            between IF and  THEN  when  no  ELSE  is  used,  are skipped.
            IF-ELSE-THEN conditionals may be nested.

IFEND       (E)
            Terminate a  conditional  interpretation  sequence  begun  by
            IFTRUE.

IFTRUE      (E)       f IFTRUE...OTHERWISE...IFEND ---
            Unlike IF..ELSE..THEN,  these  conditionals  may  be employed
            during interpretation.  In conjuction with the  words [ and ]
            they may  be  used  within  a  colon  definition  to  control
            compilation, although they  are  not  to  be compiled.  These
            words cannot be nested.

IMMED
            Mark the most recently  made dictionary entry  such that when
            encountered at compile time  it will be  executed rather than
            compiled.

INP                 m --- n
            Inputs from port m returning value n.

IOFF                      --- q
            A variable whose value is the current character offset in the
            current block used for the input string being interpreted.

J           (C)           --- m
            Within a nested DO-loop,  return the index  of the next outer
            loop.

J+                      m --- q
            Adds m to DO-loop index J. q=m+J

K           (C)           --- m
            Within a nested DO-loop, return the index of the second outer
            loop.

K+                      m --- q
            Adds m to DO-loop index K. q=m+K

LAST                      --- p
            A variable containing  the  compilation  address  of the most
            recently created dictionary entry.

LEAVE       (C)
            Force termination  of a  DO-loop at  the next  opportunity by
            setting the loop limit  equal  to  the  current  value of the
            index.  The index  itself  remains  unchanged,  and execution
            proceeds normally until LOOP or +LOOP is encountered.

LINE                    m --- p
            Leave the word address  of the begininning of  line m for the
```

block whose number is contained at BLK. (For editing purposes a block is divided into 16 lines, numbered 0-15, of 64 characters.)

LINELOAD m b ---
Begin interpreting at line m of Block b. (0 <= m <= 15)

LIST b ---
List ASCII symbolic contents of block b on the selected output device.

LIT (C) LIT m
Automatically compiled before each literal encountered in a colon definition. Execution of LIT causes the contents of the next dictionary cell to be pushed onto the stack.

LITERAL n m ---
Store n in the dictionary (as 2 words:LIT n). Does nothing if STATE is set to compile mode. If DPREC=0 then m is dropped else 3 words are compiled: DLIT n m.

LOAD b ---
Begin interpretation of block b. The block must terminate its own interpretation with ;S , --> or CONTINUED.

LOOP (C)
Increment the DO-loop index by one, terminating the loop if the new index is equal to or greater than the limit.

LOUT --- n
Leaves the address of a printer output routine on the stack.

MAX m n --- p
Leave the greater of the two numbers.

MIN m n --- p
Leave the lesser of the two numbers.

MINUS m --- -m
Negate a number by taking its two's complement.

MOD m n --- r
Leave the remainder of m/n, with the same sign as m.

MOVE p q n ---
Move the contents of n memory cells beginning at address p into n cells beginning at address q. The contents of p is moved first; overlapping of data can occur.

NAND m n --- q
Logical AND followed by COMplement.

NEXT
End of code; terminate a code definition.

NOR m n --- q
Logical OR followed by COMplement.

NOT m --- f
Equivalent to 0=

NUMBER
Convert a character string left in the dictionary buffer by WORD as a number, returning the result on the stack. The apperance of characters that cannot be properly interpreted will cause the interpreter to skip 2 instruction words.

OCTAL
Set the number-conversion base to octal.

OR m n --- q
Bitwise logical inclusive OR of m and n.

OTHERWISE (E)
An interpreter-level conditional word. See IFTRUE.

OUTP m n ---
Outputs byte-value m to output port n. The high byte of n goes out on the upper address lines for sub-port numbers.

OVER m n --- m n m
Push the second stack value.

PAGE
Clears the terminal screen or performs an action suitable to the output device currently active.

PICK n --- q
Return the nth value on the stack, not counting n itself (2 PICK is equivalent to OVER).

PRINTOUT m n ---
Lists ASCII contents of blocks n upto but not including m on selected output device. Only blocks starting with "(" are listed. The listing is prefaced by a DIR listing.

PROT
Turns on write-protection circuits in the ICEbox. Makes it impossible to write to locations below 4000H.

PUTC n ---
Outputs ASCII character n to the selected output device.

R> (C) --- n
Pop the value from the return stack and push it onto the user stack. See >R.

REPEAT (C2-,P)
Effect an unconditional jump back to the beginning of a BEGIN..WHILE..REPEAT loop. See BEGIN.

ROT m n p --- n p m
Rotate the top three values on the stack, bringing the deepest to the top.

SCR	--- q	A variable whose value is the current block used for the input string being interpreted.
SET	m p SET nnnn ---	Define a word nnnn which when executed, will cause the value m to be stored at address p.
SHOW	b ---	List ASCII symbolic contents of block b on the selected output device. Repeated pressings of the space bar on the control terminal will list the next block in sequence. Pressing any other key will terminate the sequence.
SKIP	(C)	Skips the next word within a colon definition. Used with FIND and NUMBER.
SP@	--- p	Return the address of the top of the stack. (e.g. 1 2 SP@ @ . . . would type 2 2 1)
SPACE		Output a space character to the selected output device.
SPACES	n ---	Output n spaces to the selected output device. No action for n < 1.
STATE	--- q	A variable whose value is set to compile mode or immediate mode.
SPACES?	p --- m n	Leaves starting address m and character count n of a message string beginning at address p. n is the length of the message after all trailing spaces have been subtracted starting at address p+63.
STYPE	q ---	Equivalent to COUNT TYPE.
SWAB	m --- n	Exchange the high and low order bytes of value m.
SWAP	m n --- n m	Exchange the top two stack values.
SYSCOPY	---	Copies blocks 1 thru 99 from disc drive A to drive B.
TABLE	TABLE nnnn ----	Define the beginning of a table of words. The values to be entered into the table must follow the definitions of the table. The sequence i nnnn will leave the address of the i-th word on the stack. The index should be 0 <= i < number-of-table-entries. No check is made on the range of i.

TECO
Switch CONTEXT vocabulary to TECO editor, making its definitions accessible.

TFLAG --- n
TFLAG is a variable. It is used by PUTC and EXPECT. Any printable letter output by PUTC causes TFLAG to be set to 1. If TFLAG is ZERO when EXPECT is called by the outer interpreter an OK will be printed as a prompt.

THEN (C0-,P)
Terminate an IF..ELSE..THEN conditional sequence.

TYPE m n ---
Send a string of n characters starting at byte address m to terminal.

U! m n ---
Stores value m into write protected location n and re-protects.

UERR
Undefined ERRor. Print the name at HERE and the word "undefined". Does NOT do an ABORT.

UPDATE
Flag the most recently referenced block as updated. The block will subsequently be transferred automatically to disk should its buffer be required for storage of a different block. See FLUSH.

UNPROT
Makes it possible to write to locations below 4000h in colon definitions.

VARIABLE m VARIABLE nnnn ---
Create a word nnnn which when executed will push the address of a variable (initialized to m) onto the stack.

VPTR --- q
A variable similar to DP that points to the next available variable location. Currently starts at E000h and progresses toward SP@. VPTR may be set by the user to a more useful location (i.e. C000h in commercial mode).

WHILE (C2+,P) f WHILE ---
Test the value on the stack and if FALSE exit out of a BEGIN..WHILE..REPEAT loop. See BEGIN.

WHERE
Output information about the status of Forth after an error abort. Indicate at least the last word compiled and the last block accessed.

WORD WORD pppp ---
Read the next word from the input string being interpreted until a delimiter c is found, storing the packed character string at the next available dictionary location (HERE) with

the character count in the first byte.

XOR m n --- q
Bitwise logical exclusive OR of m and n.

ZERO p ---
Set the word at location p to 0.

[(P)
Stop compilation. The words following the left bracket in a colon definition are executed, not compiled.

[[(E)
Use: [[..... f]]
Unlike BEGIN...END, these conditionals may be employed outside colon definitions. In conjuction with the words [and] they may be used within a colon definition to control compilation, although they are not compiled. These words can be nested.

] (P)
Start compilation. Following words are compiled into the dictionary.

]] (E) f ---
Terminates a conditional interpretation sequence begun by [[.

--------------------end of TERSE Glossary-------------------------------