hold "hold" PRE
(n --)
Inserts a character with ASCII value n into a pictured numeric output string. Used between **<#** and **#>**. See **#** for an example.

id. "id." PRE
(nfa --)
Displays the name of definition give its name-field address.

if "if" PRE
(flag --)
Starts a one- or two-way branch structure. Used in the form :-

: < name > ... flag if ... then ... ;

or

: < name > ... flag if ... else .. then ... ;

when <name> is executed **if** expects a flag on the stack. If the flag is nonzero the words between **if** and **else** (or the words between **if** and **then** if **else** is not used) are executed. But if the flag is zero execution branches to the words between **else** and **then** (or to the word after **then** if **else** is not used). In either case execution continues with the word after **then** . **if** ... **else** ... **then** constructs may be nested.

immediate "immediate" PRE
(--)
Marks the most recently created dictionary entry as a word that will be executed immediately even if TILE is in compile mode (rather than being compiled) as well as if tile is in execute mode.

inline "inline" P
(--)
Pauses to receive a series of characters from the current input stream, stores them in the text-input buffer and resets **IN** to point to the beginning of the received text. Input is terminated when either a carriage return or end of file character is received. In the case of an end of file character either the stream is reset to accept input form the keyboard if input was previously from a file, or TILE is exited if the input was form the keyboard.

key "key" P
(-- n)
Pauses to wait for a key to be pressed and then places the ASCII value of the key (n) on the stack. Characters received by **key** are not displayed.

last "last" PRE
(-- nfa)
Returns the name-filed address of the last definition in the **CONTEXT** vocabulary. See also **CONTEXT** , **CURRENT**.

lfa "l-f-a" PRE
(pfa -- lfa)
Given the parameter-field address of a word, returns he link-field address of that word.

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^-`abcdefghijklmnopqrstuvwxyz{\}~

lit "lit" P
(-- n)
A word that is compiled by **literal** so as to put the 16-bit value following it in the dictionary onto the stack. See also **compile**.

literal "literal" PRE , I
(n --)
Typically used in the form :-

: < name > ... [n] literal ... ;

compiles **lit** and the number (n) on the stack during compilation. When <name> is executed n is placed on the stack.

load "load" P
(--)
Used in the form :-

load < name >

loads and compiles the file specified by the space delimited string <name>.<name> must not contain any spaces and interpretation of the line is terminated after the load.

loop "loop" PRE , C
(--)
Used in the form :-

: < name > ... do ... loop ;

defines the end point of a do-loop. When <name> is executed, **do** sets the initial value of the loop index and the loop limit. Each iteration of the loop causes the loop index to be incremented by 1 when **loop** is encountered. The loop is terminated by whether the index is greater than or equal to the limit. If the loop is not terminated execution returns to just after **do**. See also **+loop**.

m* "m-times" PRE
(n1 n2 -- d)
Multiplies two single-length numbers, n1 by n2, giving the double-length product. Used if an overflow would result with *.

minus "minus" P
(n1 -- n2)
Reverses the sign of n1 to produce n2. n2 is the twos complement of n1 (i.e. zero minus n1)

move "move" PRE
(addr1 addr2 n --)
Copies n single-length (16 bit) words beginning at address addr1 to addr2. The move proceeds from low memory toward high memory. If n is zero or negated, nothing is moved. See also **cmove** .

m/ "m-divide" PRE
(d n1 -- n2)
Divides d by n1 leaving the single-length quotient n2 (rounded toward zero). An error is generated if divisor is zero.

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^-`abcdefghijklmnopqrstuvwxyz{\}~

m/mod "m-divide-mod" PRE
(d n1 -- n2 n3)
Divides d by n1 leaving the single-length remainder n2 and single-length quotient n3 (rounded toward zero). An error is generated if divisor is zero.

max "max" PRE
(n1 n2 -- n3)
Returns n3 as the greater of n1 and n2. See also **min**.

min "min" PRE
(n1 n2 -- n3)
Returns n3 as the lesser of n1 and n2. See also **max**.

mod "mod" PRE
(n1 n2 -- n3)
Divides n1 by n2 and returns the remainder n3. The quotient is rounded toward zero. An error occurs if the divisor is zero.

nfa "n-f-a" PRE
(pfa -- nfa)
Given the parameter-field address of a word, returns the name-field address of that word.

not "not" P
(n1 -- n2)
Performs a bit-by-bit negation of n1 to give n2. That is each bit in n1 is reversed in n2. Thus in binary :-

100 not

will return 1111111111111011. See also **or** , **and** , **xor** .

number "number" P
(-- n1 n2) or (-- d n2)
Converts the counted string stored in the next available dictionary location (the address returned by **here**) into a numeric representation according to the current value of **BASE**. If the string contained a '.' then the a double-length representation is returned together with a the value of 2, otherwise a single-length value if returned together with a value of 1. If the string could not be converted then a single-length 0 and False (0)are returned.

or "or" P
(n1 n2 - n3)
Performs a bit-by-bit inclusive-or of n1 with n2 to give n3. That is each bit in n1 is compared with the equivalent bit in n2 and if either or both bits are set(1) the corresponding bit in n3 will be set, otherwise the corresponding bit will be 0. Thus in binary :-

110 100 or

will return binary 110. See also **and** , **xor** .

over "over" P
(n1 n2 -- n1 n2 n1)
Copies the second number on the stack to the top of the stack.

pad "pad" PRE
(-- addr)
Returning the lowest address of a "scratchpad" area that may be used to store data temporarily. The address of **pad** is changed and the data stored there are lost if the address of the next available dictionary location (returned by **here**) is changed, that is, if anything is added to the dictionary.

pfa "p-f-a" PRE
(nfa -- pfa)
Given the name-field address of a word, returns the parameter-field address of that word.

repeat "repeat" PRE , C
(--)
Terminates a **begin** ... **while** ... **repeat** loop. Used in the form :-

: < name > ... begin ... flag while ... repeat ... ;

compiles an unconditional branch to **begin** in the definition of <name>. When <name> executes, the words between **begin** and **end** execute repeatedly while the flag on the stack remains nonzero. If the flag is False (0), execution branches to the word following **repeat**. See also **begin** , **while** , **until**.

rot "rote" P
(n1 n2 n3 -- n2 n3 n1)
Rotates the third number on the stack to the top of the stack.

s->d "single-to-double" PRE
(n -- d)
Converts a single-length number to a double-length number, retaining the correct sign.

scode "s-code" PRE
(n1 --)
Copies the single-length value on the top of the return stack (the code-field address of the word to which execution will return) into the code-field address of the most recent definition in the **CONTEXT** vocabulary. See also **;code** , **ca!**.

seal "seal" PRE
(--)
Seals of the **CONTEXT** vocabulary and prevents all previous definitions from being seen from within the vocabulary. See also **CONTEXT** , **-find**.

sign "sign" PRE
(n --)
Appends an ASCII "-" (minus sign) to the start of a pictured (formatted) numeric output string if n is negative. See **#** for an example.

smudge "smudge" PRE
(--)
Toggles a bit (the smudge bit) in the header of the most recently defined word so as to determine whether or not the word can be found during a dictionary search. If the word can be found **smudge** marks it not to be found. A second execution of **smudge** will make if "findable" again. While a word is being defined the smudge bit is set so that the word cannot be found, **;** executing **smudge** to make it "findable". This is but one of several ways to keep bad definitions from being executed.

space "space" PRE
(--)
Sends a space (blank or ASCII 32) to the console.

spaces "spaces" PRE
(n --)
Sends n spaces (blanks) to the console. Nothing is displayed if n is zero or negative.

stdin? "stdin-question" P
(-- f)
Returns single-length value of True (-1) if the standard input stream is being used otherwise a single-length value of False (0) is returned.

swap "swap" P
(n1 n2 -- n2 n1)
Swaps the top two numbers on the stack.

then "then" PRE , I , C
(--)
Marks the end of a conditional branch. Used in the form :-

: < name > ... flag if ... then ... ;

or

: < name > ... flag if ... else ... else ... then ... ;

marks where execution will continue relative to the corresponding **if** or **else**. When the **if** construct is finished, execution jumps to the word just after **then**.

toggle "toggle" PRE
(addr n1 --)
Performs the bit-by-bit toggle of the least significant byte at addr with the bits specified by the single-length mask n2 and places the least significant byte of the result back into addr. See also **(toggle)**.

type "type" PRE
(addr n --)
Displays a string of n characters from memory starting with the character at addr. Nothing is displayed if n is zero or negative. See also **count**.

u* "u-times" P
(u1 u2 -- ud)
Multiplies u1 by u2, returning the double-length product ud. All values are treated as unsigned.

u. "u-dot" PRE
(u --)
Displays the unsigned value of u at the current display position, followed by a space. See also **.** (dot).

u/ "u-divide" P
(ud1 u2 -- u3)
Divides the double-length number ud1 by u2 returning the single-length remainder u2 and the single-length quotient u3. Floored division is used. All values are treated as unsigned. An error is generated it the divisor is zero.

unlink "unlink" PRE
(--)s
Unlinks the current **CONTEXT** vocabulary and reverts to the previous **CONTEXT** vocabulary. See also **forget.**

until "until" PRE , C , I
(f --)
Marks the end of an indefinite loop. Used in the form :-

: < name > ... begin ... flag until ... ;

compiles a conditional branch back to the corresponding **begin**. When <name> is executed **until** expects a flag on the stack and execution loops back to the corresponding **begin** until the flag is nonzero.

If the flag is nonzero, execution continues with the word following **until**. See also **begin , while , repeat.**

variable "variable" PRE
(--)
A defining word that creates a single-length variable. Used in the form :-

variable < name >

creates a dictionary entry for <name> and allocates storage space for a single-length number. When <name> is executed the address of the parameter-field of <name> is placed on the stack, suitable for use by @ (fetch) and ! (store) .

vocabulary "vocabulary" PRE
(--)
A defining word that creates a vocabulary. Used in the form :-

vocabulary < name >

creates a dictionary entry for <name> that specifies a new ordered list of word definitions. After <name> is executed the vocabulary <name> will be first to be searched (the **CONTEXT** vocabulary). <name> can be made the compilation vocabulary or the vocabulary to receive new definitions (the **CURRENT** vocabulary) with :-

< name > definitions

which will cause new definitions to be appended to < names >'s path.

while "while" PRE , I , C
(f --)
Decides the continuation or termination of a **begin** ... **while** ... **repeat** loop. Used in the form :-

: < name > ... begin ... flag while ... repeat ... ;

compiles a conditional branch operator into the definition of <name>. When <name> is executed **while** expects a flag on the stack, while the flag is nonzero the words between **while** and the corresponding **repeat** are executed, and **repeat** returns execution to immediately after the corresponding **begin**. If the flag is zero, execution jumps to the word following **repeat** .

word "word" P
(n --)
Generates a counted string by nondestructively accepting characters from the input stream until a delimiting character whose ASCII code is n is encountered or until the input stream is exhausted. Does not cause execution to pause. Leading delimiters are ignored. The characters are stored as a counted string at the next available dictionary location (the value returned by **here**).

words "words" PRE
(--)
Displays a list of all words in the **CONTEXT** vocabulary.

xor "x-or" P
(n1 n2 -- n3)
Performs a bit-by-bit exclusive-or of n1 with n2 to give n3. That is, each bit of n1 is compared with the equivalent bit in n2 and if either one or the other bit is set (but not both) the corresponding bit in n3 will be set, otherwise the corresponding bit will be 0. Thus in binary :-

```
110 100 xor
```

will return 010. See also **and** , **or**.

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^-`abcdefghijklmnopqrstuvwxyz{\}~

Appendix B

TILE Memory Allocation