CHAPTER 8. COMPILER WORDS

8.1. COLON HEADER

```
:           ( -- )
```
Create a new high level dictionary word. : definitions must be terminated with a semi-colon, or aborted with a \.

SP@ HEADC E9 C, LIT DOLIST , SMUDGE

```
            HEADER      !:,Z                        ; WATCH MACRO CALL ***
COLON:      NEST
            DW          SPAT
            DW          HEADC
            DW          JMPLIT
            DW          DOLIST
            DW          SMUDGE
            DW          UNNEST
DOLIST:     ADD         AX,3
            XCHG        AX,SI
            STOSW
XNEXT:      NEXT
```

```
(CALL       ( -- )
```
A primitive operator which compiles a CALL instruction followed by the calculated offset address based on the contents of the word following the (CALL word.

R> DUP 2+ >R @ 2- HERE - ,

```
            HEADER      LLAC(,H
JSLIT:      NEST
            DW          CLIT
            DB          0E8h                        ; CALL Op code
JSCOM:      DW          CCOMM
            DW          FROMR
            DW          XDUP
            DW          TWOP
            DW          TOR
            DW          AT
            DW          TWOM
            DW          HERE
            DW          SUB
            DW          COMMA
            DW          UNNEST
```

```
(JMP        ( -- )
```
A primitive operator which compiles a machine language JMP ;instruction followed by the calculated offset address based on ;the contents of the word following the (JMP instruction.

$E9 C, R> DUP 2+ >R @ 2- HERE - ,

```
            HEADER      PMJ(,H
JMPLIT:     NEST
            DW          CLIT
            DB          0E9h                        ; JMP OP CODE
            DW          BRAN
```

```
        DW      JSCOM
```

8.2. CONSTANT AND VARIABLE

:CON (n --)
Fetch a word name from the input stream. Add this name to the dictionary as a constant with the value specified by the top of the stack. When the name is later executed, the value will be pushed on the stack.

```
        HEADER  NOC!:,Z
CON:    NEST
        DW      HEADC
        DW      JSLIT
        DW      AT
        DW      COMMA
        DW      UNNEST
```

:BUILD (--)
Used in a "Father word" to define a new "Child word". Usage is:

: father :BUILD creation logic inherited logic

Each time the "father" is executed :BUILD take following input string for the "child" name.

```
        HEADER  DLIUB!:,Z
BUILD:  NEST
        DW      HEADC
        DW      JSLIT
        DW      XNEXT
        DW      UNNEST
```

:VAR (--)
Fetch a word name from the input stream. Add this name to the dictionary as a variable, and store the value at the top of the stack as the initial value of the variable. When the name is later executed, the address of the variable will be pushed on the stack. The word @ is required to replace the address with the contents.

```
        HEADER  RAV!:,Z
VAR:    NEST
        DW      BUILD
        DW      COMMA
        DW      UNNEST
```

(DEFER (--)
Primitive for Defered words.

```
        HEADER  REFED(,H
PDEFER: MOV     BX,AX
        MOV     AX,[BX+2]
        JMP     AX
```

DEFER (--)
Make a Deferred word Creates a Deferred word. The operation of the word may be changed later by an operation such as ' name2 NEW name .

```
        HEADER  REFED,D
DEFER:  NEST
        DW      HEADC
        DW      JMPLIT
        DW      PDEFER
        DW      UNNEST
```

8.3. DOES WORDS

```
(;C           ( -- )
```
A primitive used to change from normal high-level interpretation ;to code level.
```
              HEADER     C!;(,H
PSEMIC:       NEST
              DW         FROMR
              DW         FLAST
              DW         SUB
              DW         THREE
              DW         SUB
              DW         FLAST
              DW         ONEP
              DW         STORE
              DW         UNNEST
```

```
; :           ( -- )
```
Marks where the inherited properties in a "Father-word" begin. That which follows will get executed when the "Child-word" is invoked. Used with :BUILD to create "Father-words".
```
              DB         0
              DB         ':'
              DB         ';' OR IMMFLG
              CHAIN      1B
DOES:         NEST
              DW         COMP
              DW         PSEMIC
              DW         JSLIT
              DW         DODOES
              DW         UNNEST
DODOES:       MOV        AX,SI
              STOSW
              POP        SI
              NEXT
```

8.4. COLON COMPILER WORDS

```
]:            ( -- )
```
Resume suspended : definition. The stack must be as it was when ;[was executed.
```
              HEADER     !:],1D                    ; WATCH MACRO CALL ***
SMUDGE:       NEST
              DW         STATE
              DW         SCOMP
              DW         LATEST
              DW         AT
              DW         ONEM
              DW         SCOMP
              DW         UNNEST
```

```
;             ( -- )
```
Terminate Colon definition. Check and warn if parenthesis and ;brackets don't balance.
```
              DB         0
              DB         ';' OR IMMFLG
```

```
        CHAIN   1B
SEMI:   NEST
        DW      COMP
        DW      UNNEST
        DW      QCSP
        DW      UNNEST
```

?CSP (--)
Check stack pointer to verify it is at the same place following a definition as at the beginning. This will detect unmatched looping constructs.

```
        HEADER  PSC?,1F
QCSP:   NEST
        DW      SMUDGE
        DW      SPAT
        DW      TWOP
        DW      SUB
        DW      ZBRAN
        DW      SEMI1
        DW      PTYPE
        DB      '[?]'
        DB      7,0
        DW      LIT,INPTR,AT
        DW      ZBRAN,SEMI1,BACK
SEMI1:  DW      UNNEST
```

MEM (-- n)
Gets the number of bytes of free memory available for stack and ;dictionary entries.

```
        HEADER  MEM,M
MEM:    NEST
        DW      SPAT
        DW      HERE
        .DW     SUB
        DW      UNNEST
```

INCNT (-- n)
" Input buffer count ". Return the number of characters in the circular input buffer.

```
        HEADER  TNCNI,I
INCNT:  NEST
        DW      ZERO
        DW      LIT
        DW      041Ch
        DW      XAT
        DW      ZERO
        DW      LIT
        DW      041Ah
        DW      XAT
        DW      SUB
        DW      UNNEST
```

DP (-- addr)
" Top of dictionary pointer "

```
        HEADER  PD,D
DP:     LCALL   AT
        DW      DICT
```

8.5. SYSTEM VARIABLES AND POINTERS

```
EFLAG     DB     ?                     ; Error flag for bad numbers
DPT       DB     0                     ; Flag for double numbers
ECOFLG    DW     0FFFFh                ; Echo flag for EGET
DELIM     DB     ' '                   ; Delimiter
CRSEEN    DB     0                     ; Flag for new Carriage Return
CRTXT     DB     0                     ; Flag for CR from Text buffer
BBKIV     DW     2 DUP(?)              ; Interrupt Vector for BIOS Keyboard Break
DB0IV     DW     2 DUP(?)              ; Interrupt Vector for Divide by zero
N         DW     ?                     ; Temporaries
RHOLD     DW     ?                     ;
ARGLOC    DW     ?                     ; Argument location
ARGCNT    DW     ?                     ; Character count of argument
XHOLD     DW     ?                     ; Index hold area
RAND      DW     2 DUP(?)              ;
INPTR     DW     ?                     ; Input pointer, and expansion
LBUF      DW     ?                     ; Last Buffer, offset
          DW     ?                     ; Last Buffer, segment
CBASE     DW     ?                     ; Current Base
LASTW     DW     ROOT-3                ; Most recent entry pointer
NVOCS     DW     8                     ; Maximum vocabularies in search order.
GROWNG    DW     ?                     ; Growing (Current) pointer
          DW     8 dup(0)              ; Vocabulary stack area
SRCHNG    DW     8 dup(?)              ; Searching (Context) pointer
CSTATE    DW     ?                     ; Compilation state
CSRCH     DW     ?                     ; Current vocabulary being searched.
VOCNO     DW     0                     ; Most recent vocabulary number.
VOCLINK   DW     ROOT                  ; Vocabulary link
          DW     NUMB                  ; Nominal routine for Not Found case.
          DW     BACK                  ; Nominal routine for Not a Number.
          DW     4 DUP(0)              ; Spare environment
TO        DW     TO+2                  ; Start of User variables
          DW     1130                  ; Initial Rate variable
TOES      DW     -270                  ; Top of empty stack
DICT      DW     LAST                  ; Dictionary usage top
          DW     8 DUP(?)              ;
          DB     'Copyright 13 Mar 1987 by Robert L. Smith and LaFarr Stuart'
BOTB      DW     10h                   ; Beginning of Text Buffers, Address
          DW     ?                     ; Beginning of Text, Segment
TPTR      DW     ?                     ; Text Pointer
DELCH     EQU    08h                   ; Delete character (Back-space)
TABCH     EQU    09h                   ; Tab character
BLCH      EQU    20h                   ; ASCII Space character
CRCH      EQU    0Dh                   ; ASCII Carriage Return character
LFCH      EQU    0Ah                   ; ASCII Line-Feed character
BSCH      EQU    08h                   ; ASCII Back-Space character
ESCCH     EQU    1BH                   ; ASCII Escape character
IMMFLG    EQU    80h                   ; Immediate Flag
```

8.5. POINTER WORDS

SEARCHING   Address of the vocabulary to be searched first by the interpreter.

```
          HEADER    GNIHCRAES,S
```

```
SRCH:      LCALL     AT
           DW        SRCHNG

GROWING    Address of the vocabulary to which new words are to be added.
           HEADER    GNIWORG,G
GROW:      LCALL     AT
           DW        GROWNG

VOCNUM     Address of most recent vocabulary number
           HEADER    MUNCOV,V
VOCNUM:    LCALL     AT
           DW        VOCNO

VOCTABLE             Address of the vocabulary table
           HEADER    ELBATCOV,V
VOCTAB:    LCALL     AT
           DW        VOCABT

LATEST               Address of pointer to most recent entry
           HEADER    TSETAL,L
LATEST:    LCALL     AT
           DW        LASTW
```

STATE (– addr)
Switch for compilation or execution. Execute if 0" Variable whose value is 0 when not compiling in a : definition. ;Used to allow IMMEDIATE words which behave differently when ;compiling than when simply executed from the input stream.

```
           HEADER    ETATS,S
STATE:     LCALL     AT
           DW        CSTATE
```

BASE (– addr)
Variable containing the radix for number conversions on input or ;output. Value must be greater than 1 and less than 127.

```
           HEADER    ESAB,B
BASE:      LCALL     AT
           DW        CBASE
```

ECHO (– addr)
Address of "Echo Flag" Variable yielding the address of an echo flag. If the flag is zero, the normal characters will not be echoed on input. This is useful when input has been re-directed from a file, and you do not wish to see the text displayed on the screen.

```
           HEADER    OHCE,E
ECHOF:     LCALL     AT
           DW        ECOFLG
```

USER (addr --)
"Pointer to free area" Variable which points to top of a user constant area. Typical use is: n :CON xxx 2 USER +!

```
           HEADER    RESU,U
USER:      LCALL     AT
           DW        TO
```

#VOCS (addr --)
" Address of maximum number of active vocabularies" Address of maximum number of vocabularies in search order.

```
           HEADER    SCOV#,C
NVOC:      LCALL     AT
           DW        NVOCS
```

HERE (-- addr)
This simply places the address of the first free byte above the dictionary on top.

```
        HEADER  EREH,H
HERE:   NEST
        DW      DP
        DW      AT
        DW      UNNEST
```

ALLOT (n --)
Reserve number of bytes specified by top at the end of the dictionary. Frequently used to allow space in a table following a :BUILD definition.

```
        HEADER  TOLLA,A
ALLOT:  NEST
        DW      DP
        DW      PLSTOR
        DW      UNNEST
```

, (n --)
Copy top of stack to top of the dictionary and then bump DP by 2.

```
        HEADER  !,,L
COMMA:  NEST
        DW      HERE
        DW      STORE
        DW      TWO
        DW      ALLOT
        DW      UNNEST
```

C, (char --)
Move right 8-bits from top word to top byte of dictionary, HERE. Then bump DP by one.

```
        HEADER  !,C,C
CCOMM:  NEST
        DW      HERE
        DW      CSTOR
        DW      ONE
        DW      ALLOT
        DW      UNNEST
```

COMPILE (addr -)
Copy next word in this definition into the word being compiled. This word is not IMMEDIATE; it is frequently used in words which are IMMEDIATE.

```
        HEADER  ELIPMOC,C
COMP:   NEST
        DW      QCOMP
        DW      FROMR
        DW      XDUP
        DW      TWOP
        DW      TOR
        DW      AT
        DW      COMMA
        DW      UNNEST
```

?COMP (--)
If STATE is true, i.e. compiling, do nothing. If STATE is zero issue message "CANT EXECUTE" and call QUIT.

```
        HEADER  PMOC?,1F
QCOMP:  NEST
        DW      STATE
        DW      AT
        DW      ZEQU
        DW      ZBRAN
        DW      QCOMP1
        DW      CR
        DW      PTYPE
        DB      7                       ; Ring Bell
        DB      " , Can't Execute"
        DB      0
        DW      QUIT
QCOMP1  DW      UNNEST
```

DECIMAL (--)
Sets the value of BASE to ten.

```
        HEADER  LAMICED,D
DEC:    NEST
        DW      CLIT
        DB      10
        DW      BASE
        DW      STORE
        DW      UNNEST
```