CHAPTER 3. INPUT AND OUTPUT WORDS

3.1. PRIMITIVE OUTPUT WORDS

```
ASCIIL:   SHR      AL,1           ; Convert left nibble to ASCII
          SHR      AL,1
          SHR      AL,1
          SHR      AL,1
;
ASCIIR:   AND      AX,0Fh         ; Convert right nibble to ASCII
          ADD      AL,90
          DAA
          ADC      AL,40h
          DAA
          JMP      COUT
;
P4HEX:    MOV      CL,AL          ; Print AX as 4 hex characters
          MOV      AL,AH
          CALL     P2HEX
          MOV      AL,CL
P2HEX:    MOV      CH,AL
          CALL     ASCIIL
          MOV      AL,CH
          JMP      ASCIIR
;
TYPEM:    MOV      AL,[BX]        ; Type a Null-delimited message
          OR       AL,AL
          JZ       RETN
          CALL     COUT
          INC      BX
          JMP      TYPEM
RETN:     RET
```

3.2. OUTPUT WORDS

.W   (n1 -- n1)
Hex print word, non-destructive. Display top word with 4 hex digits. No space follows and the ;value in BASE has no effect.

```
          HEADER   W.,N
HPW:      MOV      BP,SP
          MOV      AX,[BP]
          CALL     P4HEX
          JMP      SPACE
```

.B   (n1 -- n1)
Hex print byte, non-destructive. Display top, right 8-bits, with 2 hex digits. No space follows ;and value in BASE has no effect.

```
          HEADER   B.,N
HPB:      MOV      BP,SP
          MOV      AX,[BP]
          CALL     P2HEX
          JMP      SPACE
```

```
CR                      ( -- )
```
Output Carriage-Return + Line-Feed. Sends a RETURN, LINE-FEED sequence to the terminal.
```
            HEADER      RC,C
CR:         MOV         AL,CRCH
            CALL        COUT
            MOV         AL,0Ah
            CALL        COUT
            NEXT
```

```
.TYPE       ( addr -- )
```
Type a Null Delimited String
```
            HEADER      EPYT.,N
TYPED:      POP         BX
            CALL        TYPEM
            NEXT
```

```
SPACE       ( -- )
```
Print a space to the output device.
```
            HEADER      ECAPS,S
SPACE:      MOV         AX,20h
            JMP         EMIT2
```

3.3. PRIMITIVE INPUT WORDS

```
CGET:       MOV         AH,0                ; Read the real keyboard
            INT         16h                 ; Invoke BIOS Int 16h, Function 0
            AND         AL,AL               ; See if we have an extended character
            JNZ         KCTL
            MOV         AL,AH               ; Move extended character to AL,
            AND         AH,01h              ; and set bit 8.
            RET
KCTL:       XOR         AH,AH               ; Clear high byte
            RET
;
XKEYQ:      MOV         AH,1                ; See if a key is ready
            INT         16h
            JZ          NOKEY
            MOV         AX,-1
            RET
NOKEY:      XOR         AX,AX
            RET
;
RCGET:      MOV         AH,8                ; Get the next character from
            INT         21h                 ; the keyboard (or equivalent),
            XOR         AH,AH               ; using DOS Int 21h.
            RET
```

EGET
Get a character from the input device. If character is a space or above, then echo it (unless ECOFLG is 0), and return character in AL. If character is CR or TAB, echo it and set AH to 80h. If character is LF, or ESC, set AH to 80h. If character is NULL, get next character and set AH to 01. For other characters, clear AH.

```
EGET:       CALL        RCGET
            CMP         AL,' '              ; Echo space and above.
            JC          QCTL
QCOUT:      MOV         DX,ECOFLG           ; Conditionally output a character.
```

```
        OR      DX,DX
        JZ      ECLR
COUT:   MOV     DL,AL           ; Output a character in AL
        MOV     AH,2
        INT     21h
ECLR:   XOR     AH,AH           ; Clear high byte of AX.
        RET                     ; Return with character in low byte.
ESHB:   CALL    QCOUT
        MOV     AH,80h
        JMP     SHB
QCTL:   CMP     AL,LFCH         ; Line-Feed
        JE      SHB
        CMP     AL,CRCH         ; Carriage-return
        JE      ESHB
        CMP     AL,1Bh          ; Escape code
        JE      SHB
        CMP     AL,09           ; Forward Tab
        JE      ESHB
        OR      AL,AL           ; Check for null
        JNZ     ECLR
        CALL    RCGET           ; Its a NULL: get next character
        MOV     AH,01h          ; and set bit 8.
        RET
SHB:    MOV     AH,80h          ; Set high order bit
        RET
```

3.4. INPUT WORDS

KEY (-- n)
Push a character image from the user's keyboard. Read a character from the keyboard and push it with 8 leading ;zero bits as a word. There is no echo of the character. If the ;character would have been a special character, it is shifted ;right by 8 bits and bit 8 is set to a one.

```
        HEADER  YEK,K
KEY:    LJMP    KEY1
KEY1:   CALL    CGET
        AND     AH,01h
        PUSH    AX
        NEXT
```

KEY? (-- flag)
Check for any key depressed.

```
        HEADER  !?YEK,K
KEYQ:   CALL    XKEYQ
        PUSH    AX
        NEXT
```

EMIT (char --)
Output the character on the stack. Puts the low order 8 bits from top out to the terminal as a ;character.

```
        HEADER  TIME,E
EMIT:   LJMP    EMIT1
EMIT1:  POP     AX
EMIT2:  CALL    COUT
        NEXT
```