```
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO    TTTTTTTTTTTTTTT     SSSSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT         SSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBBBBBBBBBBB     OOO         OOO  OOO         OOO       TTT            SSSSSSSSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBB         BBB  OOO         OOO  OOO         OOO       TTT                     SSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
BBBBBBBBBBBB        OOOOOOOOO        OOOOOOOOO         TTT         SSSSSSSSSSS
```

```
  CCCCCCCC    000000    NN      NN    IIIIII      000000
  CCCCCCCC    000000    NN      NN    IIIIII      000000
CC           00      00  NN      NN      II     00      00
CC           00      00  NN      NN      II     00      00
CC           00      00  NNNN    NN      II     00      00
CC           00      00  NNNN    NN      II     00      00
CC           00      00  NN  NN  NN      II     00      00
CC           00      00  NN  NN  NN      II     00      00
CC           00      00  NN    NNNN      II     00      00
CC           00      00  NN    NNNN      II     00      00
CC           00      00  NN      NN      II     00      00      ....
CC           00      00  NN      NN      II     00      00      ....
  CCCCCCCC    000000    NN      NN    IIIIII      000000        ....
  CCCCCCCC    000000    NN      NN    IIIIII      000000        ....

LL             IIIIII      SSSSSSSS
LL             IIIIII      SSSSSSSS
LL               II      SS
LL               II      SS
LL               II      SS
LL               II        SSSSSS
LL               II        SSSSSS
LL               II              SS
LL               II              SS
LL               II              SS
LL               II              SS
LLLLLLLLLL     IIIIII    SSSSSSSS
LLLLLLLLLL     IIIIII    SSSSSSSS
```

Table of contents

```
                        .title  CONIO - console input output routines
                        .ident  /V04-000/
                ;
                ;*****************************************************************************
                ;*                                                                           *
                ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                  *
                ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                   *
                ;*  ALL RIGHTS RESERVED.                                                     *
                ;*                                                                           *
                ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED    *
                ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
                ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER    *
                ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY    *
                ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY    *
                ;*  TRANSFERRED.                                                             *
                ;*                                                                           *
                ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE    *
                ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT    *
                ;*  CORPORATION.                                                             *
                ;*                                                                           *
                ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS    *
                ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                  *
                ;*                                                                           *
                ;*                                                                           *
                ;*****************************************************************************
                ;
                ; Facility:  system bootstrapping
                ;
                ; Abstract: CONIO provides basic console read, readprompt and write facilities.
                ;
                ; Author:  Richard I. Hustvedt, creation date:  27-apr-1978
                ;
                ; Modified by:
                ;
                ;       V03-002 WHM0001         Bill Matthews           9-Jul-84
                ;
                ;               Add generalized alternate console terminal support
                ;
                ;       V03-001 DNC0001         David N. Cutler         29-Dec-83
                ;
                ;               Add support for QVSS as the console terminal on MicroVax I.
                ;
                ; Include files:
                ;

                        $prdef                          ; define processor registers
                        $ssdef                          ; define status code values

                ;
                ; Equated symbols:
                ;

0000000D        cr      = 13                            ; character code for carriage return
0000000A        lf      = 10                            ; character code for line feed
00000015        control_u = 21                          ; character code for control-u
00000013        control_s = 19                          ; control s (xoff)
```

```
00000011  0000    58          control_q = 17                  ; control q (xon)
0000007F  0000    59          rubout = 127                    ; character code for rubout
00000000  0000    60          v_rub  = 0                      ; rubout sequence in progress
```

```
                     0000     62          .sbttl  boo$readprompt - prompt and read input string
                     0000     63 ;+
                     0000     64 ;       boo$readprompt outputs the specified asciz prompt string on the
                     0000     65 ;       console terminal then checks the count of characters to be read.
                     0000     66 ;       If zero it exits, otherwise it reads the console terminal until
                     0000     67 ;       either a carriage return is encountered or the character count
                     0000     68 ;       is satisfied.  The specified buffer is filled with an ascic
                     0000     69 ;       string containing the characters read but not including the
                     0000     70 ;       terminating carriage return.
                     0000     71 ;
                     0000     72 ; Calling sequence:
                     0000     73 ;
                     0000     74 ;       callx   arglist,boo$readprompt
                     0000     75 ;
                     0000     76 ; Input parameters:
                     0000     77 ;
                     0000     78 ;       prompt(ap)  -  address of asciz prompt string
00000004             0000     79          prompt  =  4
                     0000     80 ;
                     0000     81 ;       size(ap)    -  maximum length of input string
00000008             0000     82          size    =  8
                     0000     83 ;                       note: if size is zero, then nothing is read
                     0000     84 ;                             and only the prompt string is written.
                     0000     85 ;
                     0000     86 ;       buf(ap)     -  address of input buffer
0000000C             0000     87          buf     =  12
                     0000     88 ;
                     0000     89 ;
                     0000     90 ; Output parameters:
                     0000     91 ;
                     0000     92 ;       r0 - completion status code (always ss$_normal)
                     0000     93 ;
                     0000     94 ;       Buffer located by buf(ap) will be filled with the string
                     0000     95 ;       read as an ascic string.
                     0000     96 ;
                     0000     97
                 00000000     98          .psect  $conio,byte
               0314  0000     99          .entry  boo$readprompt,^m<r2,r4,r8,r9>
     58  04 AC    D0  0002    100 10$:    movl    prompt(ap),r8           ;get prompt string address
         54       D4  0006    101         clrl    r4                      ;clear control flags
   50    88       9A  0008    102 20$:    movzbl  (r8)+,r0                ;get next output character
         05       13  000B    103         beql    30$                     ;if eql none
       0075       30  000D    104         bsbw    con$putchar             ;output character
         F6       11  0010    105         brb     20$                     ;
                      0012    106
     52  08 AC    9A  0012    107 30$:    movzbl  size(ap),r2             ;maximum number of characters to read
         60       13  0016    108         beql    120$                    ;if eql none
     59  0C AC    D0  0018    109         movl    buf(ap),r9              ;set address of input buffer
         89       94  001C    110         clrb    (r9)+                   ;initialize string count
         02 52    F5  001E    111         sobgtr  r2,40$                  ;decrement and test character count
         42       11  0021    112         brb     110$                    ;end of read
                      0023    113
        008F      30  0023    114 40$:    bsbw    con$getchar             ;get a character
58   50  80 8F    8B  0026    115         bicb3   #^x80,r0,r8             ;clear parity bit
58   58  7F 8F    91  002B    116         cmpb    #rubout,r8              ;rubout?
         11       12  002F    117         bneq    80$                     ;if neq no
   58    79       9A  0031    118         movzbl  -(r9),r8                ;get character to rubout
```

```
                DC   13  0034   119          beql    30$                     ;if eql none
          02 54 00   E2  0036   120          bbss    #v_rub,r4,70$           ;set start of rubout sequence
                40   10  003A   121          bsbb    outbslsh                ;output back slash
                44   10  003C   122 70$:     bsbb    outr8                   ;output rubbed out character
                52   D6  003E   123          incl    r2                      ;adjust remaining character count
                E1   11  0040   124          brb     40$                     ;
                     0042   125
          02 54 00   E5  0042   126 80$:     bbcc    #v_rub,r4,90$           ;terminate rubout sequence
                34   10  0046   127          bsbb    outbslsh                ;output backslash
             58 15   91  0048   128 90$:     cmpb    #control_u,r8           ;control u?
                B5   13  004B   129          beql    10$                     ;if eql yes
          03 58 06   E1  004D   130          bbc     #6,r8,100$              ;if clr, then graphic
             58 20   8A  0051   131          bicb    #32,r8                  ;convert to upper case
             58 0D   91  0054   132 100$:    cmpb    #cr,r8                  ;carriage return?
                0C   13  0057   133          beql    110$                    ;if eql yes
                52   D5  0059   134          tstl    r2                      ;any space left in buffer?
                C6   13  005B   135          beql    40$                     ;if eql no
                23   10  005D   136          bsbb    outr8                   ;echo character
             89 58   90  005F   137          movb    r8,(r9)+                ;buffer new character
               BE 52   F4  0062   138          sobgeq  r2,40$                  ;reduce space remaining (always loop)
                     0065   139
             58 0D   9A  0065   140 110$:    movzbl  #cr,r8                  ;set carriage return character
                1B   10  0068   141          bsbb    con$putchar             ;
             50 0A   9A  006A   142          movzbl  #lf,r0                  ;yes send line feed also
                16   10  006D   143          bsbb    con$putchar             ;output character in r0
          59   0C AC   C2  006F   144          subl    buf(ap),r9              ;compute character count + 1
    0C BC 59   01   83  0073   145          subb3   #1,r9,@buf(ap)          ;set actual character count
             50 01   3C  0078   146 120$:    movzwl  #ss$_normal,r0          ;return normal completion status
                04   007B   147          ret                             ;
                     007C   148
                     007C   149 outbslsh:                                 ;output back slash
          50   5C 8F   9A  007C   150          movzbl  #^a%\%,r0               ;set character code
                03   11  0080   151          brb     con$putchar                     ;and output it
                     0082   152
             50 58   9A  0082   153 outr8:    movzbl  r8,r0                   ;get character to output
                     0085   154
                     0085   155 con$putchar::                               ;output character in r0
                0000 31  0085   156          brw     w^10$                   ;console terminal output vector
                     0088   157
             51 20   DB  0088   158 10$:     mfpr    #pr$_rxcs,r1            ;receiver ready?
          1B 51 07   E1  008B   159          bbc     #7,r1,30$               ;if clr, receiver not ready
             51 21   DB  008F   160          mfpr    #pr$_rxdb,r1            ;read input character.
    13    51 07   00  ED  0092   161          cmpzv   #0,#7,r1,#control_s     ;control-s?
                11   12  0097   162          bneq    30$                     ;if neq no
             51 20   DB  0099   163 20$:     mfpr    #pr$_rxcs,r1            ;receiver ready?
          F9 51 07   E1  009C   164          bbc     #7,r1,20$               ;if clr, receiver not ready
             51 21   DB  00A0   165          mfpr    #pr$_rxdb,r1            ;read input character
    11    51 07   00  ED  00A3   166          cmpzv   #0,#7,r1,#control_q     ;is it a control-q?
                EF   12  00A8   167          bneq    20$                     ;no, wait for another character.
             51 22   DB  00AA   168 30$:     mfpr    #pr$_txcs,r1            ;transmitter done?
          F9 51 07   E1  00AD   169          bbc     #7,r1,30$               ;if clr, transmitter not done
             23 50   DA  00B1   170          mtpr    r0,#pr$_txdb            ;write output character
                05   00B4   171          rsb                             ;return
                     00B5   172
                     00B5   173
                     00B5   174 con$getchar::
                0000 51  00B5   175          brw     w^10$                   ;console terminal input vector
```

```
                     00B8   176
    50   20   DB     00B8   177 10$:    mfpr    #pr$_rxcs,r0            ;receiver ready?
F9  50   07   E1     00BB   178         bbc     #7,r0,10$               ;if clr, receiver not ready
    50   21   DB     00BF   179         mfpr    #pr$_rxdb,r0            ;read input character
               05    00C2   180         rsb                             ;return
                     00C3   181
                     00C3   182 con$owncty::                            ;these routines are noops in SYSBOOT
                     00C3   183 con$releasecty::
               05    00C3   184         rsb
                     00C4   185
                     00C4   186         .end
```

```
BOO$READPROMPT                  00000000 RG    02
BUF                           = 0000000C
CON$GETCHAR                     000000B5 RG    02
CON$OWNCTY                      000000C3 RG    02
CON$PUTCHAR                     00000085 RG    02
CON$RELEASECTY                  000000C3 RG    02
CONTROL_Q                     = 00000011
CONTROL_S                     = 00000013
CONTROL_U                     = 00000015
CR                            = 0000000D
LF                            = 0000000A
OP$_ACBD                      = 0000006F
OP$_ACBF                      = 0000004F
OP$_ACBG                      = 00004FFD
OP$_ACBH                      = 00006FFD
OP$_ADDD2                     = 00000060
OP$_ADDD3                     = 00000061
OP$_ADDF2                     = 00000040
OP$_ADDF3                     = 00000041
OP$_ADDG2                     = 000040FD
OP$_ADDG3                     = 000041FD
OP$_ADDH2                     = 000060FD
OP$_ADDH3                     = 000061FD
OP$_ADDP4                     = 00000020
OP$_ADDP6                     = 00000021
OP$_ASHP                      = 000000F8
OP$_CLRD                      = 0000007C
OP$_CLRF                      = 000000D4
OP$_CLRG                      = 0000007C
OP$_CLRH                      = 00007CFD
OP$_CMPD                      = 00000071
OP$_CMPF                      = 00000051
OP$_CMPG                      = 000051FD
OP$_CMPH                      = 000071FD
OP$_CMPP3                     = 00000035
OP$_CMPP4                     = 00000037
OP$_CRC                       = 0000000B
OP$_CVTBD                     = 0000006C
OP$_CVTBF                     = 0000004C
OP$_CVTBG                     = 00004CFD
OP$_CVTBH                     = 00006CFD
OP$_CVTDB                     = 00000068
OP$_CVTDF                     = 00000076
OP$_CVTDH                     = 000032FD
OP$_CVTDL                     = 0000006A
OP$_CVTDW                     = 00000069
OP$_CVTFB                     = 00000048
OP$_CVTFD                     = 00000056
OP$_CVTFG                     = 000099FD
OP$_CVTFH                     = 000098FD
OP$_CVTFL                     = 0000004A
OP$_CVTFW                     = 00000049
OP$_CVTGB                     = 000048FD
OP$_CVTGF                     = 000033FD
OP$_CVTGH                     = 000056FD
OP$_CVTGL                     = 00004AFD
OP$_CVTGW                     = 000049FD
OP$_CVTHB                     = 000068FD
OP$_CVTHD                     = 0000F7FD
OP$_CVTHF                     = 0000F6FD
OP$_CVTHG                     = 000076FD
OP$_CVTHL                     = 00006AFD
OP$_CVTHW                     = 000069FD
OP$_CVTLD                     = 0000006E
OP$_CVTLF                     = 0000004E
OP$_CVTLG                     = 00004EFD
OP$_CVTLH                     = 00006EFD
OP$_CVTLP                     = 000000F9
OP$_CVTPL                     = 00000036
OP$_CVTPS                     = 00000008
OP$_CVTPT                     = 00000024
OP$_CVTRDL                    = 0000006B
OP$_CVTRFL                    = 0000004B
OP$_CVTRGL                    = 00004BFD
OP$_CVTRHL                    = 00006BFD
OP$_CVTSP                     = 00000009
OP$_CVTTP                     = 00000026
OP$_CVTWD                     = 0000006D
OP$_CVTWF                     = 0000004D
OP$_CVTWG                     = 00004DFD
OP$_CVTWH                     = 00006DFD
OP$_DIVD2                     = 00000066
OP$_DIVD3                     = 00000067
OP$_DIVF2                     = 00000046
OP$_DIVF3                     = 00000047
OP$_DIVG2                     = 000046FD
OP$_DIVG3                     = 000047FD
OP$_DIVH2                     = 000066FD
OP$_DIVH3                     = 000067FD
OP$_DIVP                      = 00000027
OP$_EDITPC                    = 00000038
OP$_EMODD                     = 00000074
OP$_EMODF                     = 00000054
OP$_EMODG                     = 000054FD
OP$_EMODH                     = 000074FD
OP$_MATCHC                    = 00000039
OP$_MNEGD                     = 00000072
OP$_MNEGF                     = 00000052
OP$_MNEGG                     = 000052FD
OP$_MNEGH                     = 000072FD
OP$_MOVD                      = 00000070
OP$_MOVF                      = 00000050
OP$_MOVG                      = 000050FD
OP$_MOVH                      = 000070FD
OP$_MOVP                      = 00000034
OP$_MOVTC                     = 0000002E
OP$_MOVTUC                    = 0000002F
OP$_MULD2                     = 00000064
OP$_MULD3                     = 00000065
OP$_MULF2                     = 00000044
OP$_MULF3                     = 00000045
OP$_MULG2                     = 000044FD
OP$_MULG3                     = 000045FD
OP$_MULH2                     = 000064FD
```

```
OP$_MULH3                       = 000065FD
OP$_MULP                        = 00000025
OP$_POLYD                       = 00000075
OP$_POLYF                       = 00000055
OP$_POLYG                       = 000055FD
OP$_POLYH                       = 000075FD
OP$_SCANC                       = 0000002A
OP$_SKPC                        = 0000003B
OP$_SPANC                       = 0000002B
OP$_SUBD2                       = 00000062
OP$_SUBD3                       = 00000063
OP$_SUBF2                       = 00000042
OP$_SUBF3                       = 00000043
OP$_SUBG2                       = 000042FD
OP$_SUBG3                       = 000043FD
OP$_SUBH2                       = 000062FD
OP$_SUBH3                       = 000063FD
OP$_SUBP4                       = 00000022
OP$_SUBP6                       = 00000023
OP$_TSTD                        = 00000073
OP$_TSTF                        = 00000053
OP$_TSTG                        = 000053FD
OP$_TSTH                        = 000073FD
OUTBSLSH                          0000007C R      02
OUTR8                             00000082 R      02
PR$_RXCS                        = 00000020
PR$_RXDB                        = 00000021
PR$_TXCS                        = 00000022
PR$_TXDB                        = 00000023
PROMPT                          = 00000004
RUBOUT                          = 0000007F
SIZE                            = 00000008
SS$_NORMAL                      = 00000001
V_ROB                           = 00000000
```

Psect synopsis

PSECT name	Allocation		PSECT No.	Attributes
. ABS .	00000000	(0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
ABS	00000000	(0.)	01 (1.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
$CONIO	000000C4	(196.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	38	00:00:00.07	00:00:00.39
Command processing	133	00:00:00.79	00:00:04.76
Pass 1	458	00:00:12.51	00:00:24.90
Symbol table sort	0	00:00:01.33	00:00:02.51
Pass 2	54	00:00:03.68	00:00:06.87
Symbol table output	18	00:00:00.14	00:00:00.34

```
Psect synopsis output              2     00:00:00.02    00:00:00.02
Cross-reference output             0     00:00:00.00    00:00:00.00
Assembler run totals             705     00:00:18.55    00:00:39.80

The working set limit was 1650 pages.
59843 bytes (117 pages) of virtual memory were used to buffer the intermediate code.
There were 50 pages of symbol table space allocated to hold 865 non-local and 14 local symbols.
2938 source lines were read in Pass 1, producing 16 object records in Pass 2.
136 pages of virtual memory were used to define 135 macros.
```

Macro library statistics

Macro library name	Macros defined
_255DUA28:[BOOTS.OBJ]BOOTS.MLB;1	0
_255DUA28:[SYS.OBJ]LIB.MLB;1	0
_255DUA28:[SYSLIB]STARLET.MLB;2	6
TOTALS (all libraries)	6

```
918 GETS were required to define 6 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:CONIO/OBJ=OBJ$:CONIO MASD$:[EMULAT.SRC]MISSING/UPDATE=(MASD$:[EMULAT.ENH]MISSING)+MASD$:[BOOTS.SRC]CONIO/UPDATE=(MASD
```

CVBTDRIVR
LIS

DDBTDRIVR
LIS

DMBTDRIVR
LIS

DQBTDRIVR
LIS

CONIO
LIS

LOADERSUB
LIS

CONFIGUTL
LIS

DLBTDRIVR
LIS

DXBTDRIVR
LIS

INITPGFIL
LIS

LOCKDATA
LIS

LOADDRIV
LIS

LOADER
LIS