

[illegible]

[illegible]


```

0000 1      .TITLE SWITCH TERMINAL - Switch terminal port to or from DECNET
0000 2      .IDENT /V04-000/
0000 3
0000 4      *****
0000 5      *
0000 6      *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 7      *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 8      *  ALL RIGHTS RESERVED.
0000 9      *
0000 10     *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 11     *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 12     *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 13     *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 14     *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 15     *  TRANSFERRED.
0000 16     *
0000 17     *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 18     *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 19     *  CORPORATION.
0000 20     *
0000 21     *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 22     *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 23     *
0000 24     *
0000 25     *****
0000 26
0000 27     ++
0000 28     FACILITY: SET command
0000 29
0000 30     ABSTRACT:
0000 31
0000 32     This module supports switching a terminal port from terminal line
0000 33     to asynch DDCMP mode and vice versa.
0000 34
0000 35     AUTHOR:
0000 36
0000 37     Meg Dumont 1-Feb-84
0000 38
0000 39     MODIFIED BY:
0000 40
0000 41     V03-003 MMD0327      Meg Dumont,      31-Aug-1984  17:45
0000 42     Fix to clear UCBSL_FR4 field when swtiching back to a terminal
0000 43
0000 44     V03-002 MMD0319      Meg Dumont,      25-Jul-1984  15:38
0000 45     Fix to call IOCSVERFIYCHAN before raising our IPL.
0000 46
0000 47     V03-001 TMH0001      Tim Halvorsen      18-Mar-1984
0000 48     Fix to lock down code pages before executing high IPL code.
0000 49     --

```

```
0000 51
0000 52 $ccbdef
0000 53 $devdef
0000 54 $ddbdef
0000 55 $dptdef
0000 56 $ipldef
0000 57 $jpidf
0000 58 $prvdef
0000 59 $ssdef
0000 60 $ttdef
0000 61 $ttyucbdef
0000 62 $ttyvecdef
0000 63 $ttymodem
0000 64 $ucbdef
0000 65
0000 66
0000 67 : temp
00000000 68 .PSECT SET$RWDATA,NOEXE, LONG
0000 69
52 45 56 49 52 44 4F 4E 00' 0000 70 NODRV_NAME: .ASCIC /NODRIVER/
08 0000
00009 71
00000000 72 .PSECT SET$CODE,NOWRT
0000 73
0000 74 :++
0000 75 :SWITCH_TO_DDCMP
0000 76 :
0000 77 : This routine does the actual switch. First using the channel assigned
0000 78 : we find out the virtual UCB for this device. Next
0000 79 : we check the reference count in the virtual UCB to ensure it is not
0000 80 : owned by anyone else. It does this by subtracting 1 from the reference
0000 81 : count field. SET TERMINAL has a channel assigned to the device so we can
0000 82 : account for one of the channels. If any other channels are assigned
0000 83 : we can not allow the switch to occur. Please note that if this truly a
0000 84 : virtual UCB (ie. VTA0 variety), the reference count in the UCB will be
0000 85 : greater than one and the switch will fail. We check to see if the
0000 86 : ASYNCH DDCMP CLASS driver, NODRIVER is loaded. Next we
0000 87 : set the appropriate fields in the UCB to point to the NODRIVER.
0000 88 : We also set the NET bit in the DEVCHAR field to reflect that this
0000 89 : is an asynch ddmcp line. Finally , we call the port at its
0000 90 : ABORT and RESUME entry points to ensure that there is nothing outstanding
0000 91 : on the line.
0000 92 :
0000 93 : INPUTS:
0000 94 : 4(ap) = Channel assigned to device
0000 95 :
0000 96 : OUTPUTS:
0000 97 : R0 = Status of the switch
0000 98 : $$$_NORMAL - Switch was made
0000 99 : $$$_IVCHAN - Invalid channel
0000 100 : $$$_NOSUCHDEV - NODRIVER was not loaded
0000 101 : $$$_DEVALLOC - Device was in use no switch is made
0000 102 :
0000 103 :--
0000 104 switch_to_ddcmp::
0000 105 .word 0
3E 0000 0000 106 pushr #^m<r1,r2,r3,r4,r5>
BB 0002
```



```
50 04 AC D0 0004 107      movl 4(ap),r0      ; get the channel number
00000000'GF 16 0008 108      jsb g^ioc$verifychan ; Call to get CCB from channel
000E 109      setipl switch_ddcmp_ipl ; Disable for synch to IODATA BASE
0015 110      ; lock down this code
50 013C 8F B1 0015 111      cmpw #ss$ivchan,r0 ; If eql then not proper channel
69 13 001A 112      beql switch_err
51 55 61 D0 001C 113      movl ccb$l_ucb(r1),r5 ; Get the virtual UCB for device
5C A5 01 A3 001F 114      subw3 #1,ucb$w_refc(r5),r1 ; If NEQ then channels assign
65 12 0024 115      bneq switch_ddcmp_err_inuse ; do not allow switch
50 00A0 C5 D0 0026 116      movl ucb$l_fl_phyucb(r5),r0 ; Get the physical UCB
03 13 002B 117      beql 10$ ; If eql field not setup
55 50 D0 002D 118      movl r0,r5
0030 119 10$: setipl ucb$b_dipl(r5) ; Disable device interrupts
50 00000000'GF D0 0034 120      movl g^no$gl_dpt,r0 ; Get the address of the Nodriver dp
59 13 003B 121      beql switch_ddcmp_notloaded ; If eql driver not loaded
51 1E A0 3C 003D 122      movzwl dpt$w_vector(r0),r1 ; Get the offset of the class vector
50 51 C0 0041 123      addl2 r1,r0 ; Get the addr of the class vector a
0114 C5 50 D0 0044 124      movl r0,ucb$l_tt_class(r5) ; Set asynch ddcmp vector address
010C C5 60 D0 0049 125      movl class_getnxt(r0),ucb$l_tt_getnxt(r5) ; Set Asynch ddcmp getnxt
0110 C5 04 A0 D0 004E 126      movl class_putnxt(r0),ucb$l_tt_putnxt(r5) ; Set asynch ddcmp putnxt
0088 C5 10 A0 D0 0054 127      movl class_ddt(r0),ucb$l_ddt(r5) ; Set DDT address
00A8 C5 7C 005A 128      clrq ucb$q_tl_brkthru(r5) ; Clear class drivers dependant fie
51 28 A5 D0 005E 129      movl ucb$l_ddb(r5),r1 ; Get DDB
0C A1 51 D0 0062 130      movl r1,ddb$l_ddt(r1) ; Reset the DDT
38 A5 2000 8F A8 0066 131      bisw2 #dev$m_net,ucb$l_devchar(r5) ; Set that this is a network device
50 0118 C5 D0 006C 132      movl ucb$l_ft_port(r5),r0 ; Get port vector table
20 B0 16 0071 133      jsb @port_abort(r0) ; Call abort to clear line
50 0118 C5 D0 0074 134      movl ucb$l_tt_port(r5),r0 ; Get port vector table
24 B0 16 0079 135      jsb @port_resume(r0) ; Call resume to restart line
007C 136      setipl #0
007F 137      popr #^m<r1,r2,r3,r4,r5>
50 01 3C 0081 138      movzwl #ss$_normal,r0
04 0084 139      ret
0085 140
0085 141 switch_err:
0085 142      setipl #0
3E BA 0088 143      popr #^m<r1,r2,r3,r4,r5>
04 008A 144      ret
008B 145
008B 146 switch_ddcmp_err_inuse:
008B 147      setipl #0
008E 148      popr #^m<r1,r2,r3,r4,r5>
50 0840 8F 3C 0090 149      movzwl #ss$_devalloc,r0
04 0095 150      ret
0096 151
0096 152 switch_ddcmp_notloaded:
0096 153      setipl #0
0099 154      popr #^m<r1,r2,r3,r4,r5>
50 0908 8F 3C 009B 155      movzwl #ss$_nosuchdev,r0
04 00A0 156      ret
00A1 157
00A1 158 switch_ddcmp_ipl:
00A1 159      .long ipl$_synch
00A5 160
00A5 161 ;++
00A5 162 ;:SWITCH_TO_TERMINAL
00A5 163 ;
```

```
00A5 164 : This routine does the actual switch. First using the channel assigned
00A5 165 : we find out the virtual UCB for this device. Next
00A5 166 : we check the reference count in the virtual UCB to ensure it is not
00A5 167 : owned by anyone else. It does this by subtracting 1 from the reference
00A5 168 : count field. SET TERMINAL has a channel assigned to the device so we can
00A5 169 : account for one of the channels. If any other channels are assigned
00A5 170 : we can not allow the switch to occur. Please note that if this truly a
00A5 171 : virtual UCB (ie. VTA0 variety), the reference count in the UCB will be
00A5 172 : greater than one and the switch will fail. Next we
00A5 173 : set the appropriate fields in the UCB to point to the terminal class
00A5 174 : driver. We clear the NET bit in DEVCHAR to reflect that this
00A5 175 : is once again a terminal. It will also call the port at its ABORT and RESUME
00A5 176 : entry ; points to ensure that there is nothing outstanding on the line. We
00A5 177 : must also ensure that the terminal is put back to normal which means
00A5 178 : calling the terminal class driver at its SETUP UCB entry point to reset
00A5 179 : the CLASS dependant portion of the UCB, and calling the PORT driver at
00A5 180 : its DS_TRANS entry point to ensure that the modem signals were reset properly.
00A5 181 :
00A5 182 : INPUTS:
00A5 183 : 4(ap) = Channel assigned to device
00A5 184 :
00A5 185 : OUTPUTS:
00A5 186 : R0 = Status of the switch
00A5 187 : $$$_NORMAL - Switch was made
00A5 188 : $$$_DEVALLOC - Device was in use no switch is made
00A5 189 :
00A5 190 :--
00A5 191 switch_to_terminal::
00A5 192 .word 0
00A7 193 pushr #^m<r1,r2,r3,r4,r5>
00A9 194 movl 4(ap),r0 ; get the channel number
00AD 195 jsb g^ioc$verifychan ; Call to get CCB from channel
00B3 196 setipl switch_terminal_ipl ; Disable for synch to IODATA BASE
00BA 197 ; & lock down this code
00BA 198 cmpw #ss$ivchan,r0 ; If eql then not proper channel
00BF 199 beql switch_err
00C1 200 movl ccb$l_ucb(r1),r5 ; Get the virtual UCB for device
00C4 201 subw3 #1,ucb$w_refc(r5),r1 ; If EQL then no channels assign
00C9 202 beql 5$ ; allow switch
00CB 203 brw switch_terminal_err_inuse ; Else do not allow switch
00CE 204 5$: movl ucb$l_el_phyucb(r5),r0 ; Get the physical UCB
00D3 205 beql 10$ ; If eql field not setup
00D5 206 movl r0,r5
00D8 207 10$: setipl ucb$b_dipl(r5) ; Disable device interrupts
00DC 208 movl g^tty$gl_dpt,r0 ; Get the address of the TIdriver dp
00E3 209 movzwl dpt$w_vector(r0),r1 ; Get the offset of the class vector
00E7 210 addl2 r1,r0 ; Get the addr of the class vector a
00EA 211 movl r0,ucb$l_tt_class(r5) ; Set terminal class vector address
00EF 212 movl class_getnxt(r0),ucb$l_tt_getnxt(r5) ; Set terminal class getnxt
00F4 213 movl class_putnxt(r0),ucb$l_tt_putnxt(r5) ; Set terminal class putnxt
00FA 214 movl class_ddt(r0),ucb$l_ddt(r5) ; Set DDT address
0100 215 clrq ucb$q_tl_brkthru(r5) ; Clear class drivers dependant fie
0104 216 clrl ucb$l_fr4(r5) ; Clear FR4 because the class port
0107 217 ; interface uses this for fork info
0107 218 movl ucb$l_ddb(r5),r1 ; Get DDB
010B 219 movl r1,ddb$l_ddt(r1) ; Reset the DDT
010F 220 bicw2 #dev$m_net,ucb$l_devchar(r5) ; Clear no longer a network device
```

50 04 AC BB 0000 00000000'GF 16

50 013C 8F B1 00BA 198

51 5C A5 01 A3 00C4 201

50 00A0 C5 D0 00CE 204

50 00000000'GF D0 00DC 208

51 1E A0 3C 00E3 209

0114 C5 50 D0 00EA 211

010C C5 60 D0 00EF 212

0110 C5 04 A0 D0 00F4 213

0088 C5 10 A0 D0 00FA 214

00A8 C5 7C 0100 215

14 A5 D4 0104 216

51 28 A5 D0 0107 217

OC A1 51 D0 010B 219

38 A5 2000 8F AA 010F 220


```
42 A5 00F1 C5 B0 0115 221      movw    ucb$w_tt_desize(r5),ucb$w_devbufsiz(r5) ; Reset page size
50 0118 C5 D0 011B 222      movl    ucb$l_tt_port(r5),r0 ; Get port vector table
    20 B0 16 0120 223      jsb      @port_abort(r0) ; Call abort to clear line
50 0118 C5 D0 0123 224      movl    ucb$l_tt_port(r5),r0 ; Get port vector table
    24 B0 16 0128 225      jsb      @port_resume(r0) ; Call resume to restart line
50 0114 C5 D0 012B 226      movl    ucb$l_tt_class(r5),r0 ; Get the class vector table
    08 B0 16 0130 227      jsb      @class_setup_ucb(r0) ; Reset the ucb for terminal driver
50 0118 C5 D0 0133 228      movl    ucb$l_tt_port(r5),r0 ; Get port vector table
    08 B0 16 0138 229      jsb      @port_set_line(r0) ; Call resume to restart line
OB 44 A5 15 E1 013B 230      bbc      #tt$v_modem,ucb$l_devdepend(r5),20$
    51 00 9A 0140 231      movzbl   #modem$c_init,r1
50 0114 C5 D0 0143 232      movl    ucb$l_tt_class(r5),r0 ; Get the class vector table
    0C B0 16 0148 233      jsb      @class_ds_tran(r0) ; Reset the modem control
    3E BA 014B 234 20$:    setipl   #0
    01 3C 014E 235      popr      #^m<r1,r2,r3,r4,r5>
    01 3C 0150 236      movzwl   #ss$_normal,r0
    04 0153 237      ret
    0154 238
    0154 239 switch_terminal_err_inuse:
    0154 240      setipl   #0
    3E BA 0157 241      popr      #^m<r1,r2,r3,r4,r5>
50 0840 8F 3C 0159 242      movzwl   #ss$_devalloc,r0
    04 015E 243      ret
    015F 244
    015F 245 switch_terminal_ipl:
00000008 015F 246      .long    ipl$_synch ; Used to fault in code &
    0163 247 ; lock down pages simultaneously
    0163 248      .END
```

SWITCH_TERMINAL
Symbol table

- Switch terminal port to or from DECNET N 13
15-SEP-1984 23:49:36 VAX/VMS Macro V04-00 Page 6
6-SEP-1984 11:30:49 [CLIUTL.SRC]SWITCHTRM.MAR;1 (3)

```
CCBSL_UCB      = 00000000
CLASS_DDT      = 00000010
CLASS_DS_TRAN  = 0000000C
CLASS_GETNXT   = 00000000
CLASS_PUTNXT   = 00000004
CLASS_SETUP_UCB = 00000008
DDBSL_DDT      = 0000000C
DEVSM_NET      = 00002000
DPTSW_VECTOR   = 0000001E
IOCSVERIFYCHAN ***** X 03
IPLS_SYNCH     = 00000008
MODEMSC_INIT   = 00000000
NOSGL_DPT      ***** X 03
NODRV_NAME     = 00000000 R 02
PORT_ABORT     = 00000020
PORT_RESUME    = 00000024
PORT_SET_LINE  = 00000008
PRS_IPL        ***** X 03
SSS_DEVALLOC   = 00000840
SSS_IVCHAN     = 0000013C
SSS_NORMAL     = 00000001
SSS_NOSUCHDEV  = 00000908
SWITCH_DDCMP_ERR_INUSE = 0000008B R 03
SWITCH_DDCMP_IPL = 000000A1 R 03
SWITCH_DDCMP_NOTLOADED = 00000096 R 03
SWITCH_ERR     = 00000085 R 03
SWITCH_TERMINAL_ERR_INUSE = 00000154 R 03
SWITCH_TERMINAL_IPL = 0000015F R 03
SWITCH_TO_DDCMP = 00000000 RG 03
SWITCH_TO_TERMINAL = 000000A5 RG 03
TTSV_MODEM     = 00000015
TTYSGL_DPT     ***** X 03
UCBSB_DIPL     = 0000005E
UCBSL_DDB      = 00000028
UCBSL_DDT      = 00000088
UCBSL_DEVCHAR  = 00000038
UCBSL_DEVDEPEND = 00000044
UCBSL_FR4      = 00000014
UCBSL_TL_PHYUCB = 000000A0
UCBSL_TT_CLASS = 00000114
UCBSL_TT_GETNXT = 0000010C
UCBSL_TT_PORT  = 00000118
UCBSL_TT_PUTNXT = 00000110
UCBSQ_TL_BRKTHRU = 000000A8
UCBSW_DEVBUFSIZ = 00000042
UCBSW_REFC     = 0000005C
UCBSW_TT_DESIZE = 000000F1
```

+-----+
! Psect synopsis !
+-----+

PSECT name	Allocation		PSECT No.	Attributes													
ABS	00000000	(0.)	00	(0.)	NOPI	USR	CON	ABS	LCL	NOSHR	NOEXE	NORD	NOWRT	NOVEC	BYTE		
\$ABSS	00000000	(0.)	01	(1.)	NOPI	USR	CON	ABS	LCL	NOSHR	EXE	RD	WRT	NOVEC	BYTE		
SET\$RWDATA	00000009	(9.)	02	(2.)	NOPI	USR	CON	REL	LCL	NOSHR	NOEXE	RD	WRT	NOVEC	LONG		

SWITCH TERMINAL
Psect synopsis

B 14
- Switch terminal port to or from DECNET 15-SEP-1984 23:49:36 VAX/VMS Macro V04-00 Page 7
6-SEP-1984 11:30:49 [CLIUTL.SRC]SWITCHTRM.MAR;1 (3)

SET\$CODE

00000163 (355.) 03 (3.) NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	11	00:00:00.09	00:00:00.71
Command processing	81	00:00:00.89	00:00:05.57
Pass 1	403	00:00:14.38	00:00:44.52
Symbol table sort	0	00:00:02.50	00:00:08.61
Pass 2	74	00:00:02.28	00:00:08.97
Symbol table output	7	00:00:00.09	00:00:00.09
Psect synopsis output	2	00:00:00.03	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	581	00:00:20.26	00:01:08.50

The working set limit was 1350 pages.
81109 bytes (159 pages) of virtual memory were used to buffer the intermediate code.
There were 90 pages of symbol table space allocated to hold 1633 non-local and 4 local symbols.
248 source lines were read in Pass 1, producing 16 object records in Pass 2.
23 pages of virtual memory were used to define 22 macros.

! Macro library statistics !

Macro library name	Macros defined
-----	-----
-\$255\$DUA28:[CLIUTL.OBJ]CLIUTL.MLB;1	0
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	10
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	8
TOTALS (all libraries)	18

1744 GETS were required to define 18 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:SWITCHTRM/OBJ=OBJ\$:SWITCHTRM MSRC\$:SWITCHTRM/UPDATE=(ENH\$:SWITCHTRM)+EXECMLS/LIB+LIB\$:CLIUTL/LIB

0059

AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY