


```

RRRRRRR      EEEEEEEEE  AAAAAA  DDDDDDD  LL      BBBB      NN      NN
RRRRRRR      EEEEEEEEE  AAAAAA  DDDDDDD  LL      BBBB      NN      NN
RR      RR    EE          AA      AA  DD      DD  LL      BB      NN      NN
RR      RR    EE          AA      AA  DD      DD  LL      BB      NN      NN
RR      RR    EE          AA      AA  DD      DD  LL      BB      NN      NN
RRRRRRR      EEEEEEEEE  AA      AA  DD      DD  LL      BB      NN      NN
RRRRRRR      EEEEEEEEE  AA      AA  DD      DD  LL      BB      NN      NN
RR  RR      EE          AAAAAAAAAA DD      DD  LL      BB      NN      NN
RR  RR      EE          AAAAAAAAAA DD      DD  LL      BB      NN      NN
RR      RR    EE          AA      AA  DD      DD  LL      BB      NN      NN
RR      RR    EE          AA      AA  DD      DD  LL      BB      NN      NN
RR      RR    EE          AA      AA  DD      DD  LL      BB      NN      NN
RR      RR    EEEEEEEEE  AA      AA  DDDDDDD  LLLLLLLLL  BBBB      NN      NN
RR      RR    EEEEEEEEE  AA      AA  DDDDDDD  LLLLLLLLL  BBBB      NN      NN

```

```

....
....
....
....

```

```

LL      IIIIII  SSSSSSS
LL      IIIIII  SSSSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SSSSS
LL      II      SSSSS
LL      II      SS
LL      II      SS
LL      II      SS
LL      II      SS
LLLLLLLL  IIIIII  SSSSSSS
LLLLLLLL  IIIIII  SSSSSSS

```

(2) 58
(3) 127

RTF\$TARGET_DEV - Records the name of Target device.
FIL\$READ_LBN - Routine to read a logical block.

```

0000 1      .TITLE READLBN - WRITEBOOT-RTFILREAD I/O interface.
0000 2      .IDENT 'V04-000'
0000 3
0000 4
0000 5 :*****
0000 6 :*
0000 7 :* COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0000 8 :* DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0000 9 :* ALL RIGHTS RESERVED.
0000 10 :*
0000 11 :* THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0000 12 :* ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0000 13 :* INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0000 14 :* COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0000 15 :* OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0000 16 :* TRANSFERRED.
0000 17 :*
0000 18 :* THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0000 19 :* AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0000 20 :* CORPORATION.
0000 21 :*
0000 22 :* DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0000 23 :* SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0000 24 :*
0000 25 :*
0000 26 :*****
0000 27
0000 28 ++
0000 29
0000 30 FACILITY:
0000 31
0000 32 Provide an interface to allow RTFILREAD to work in the WRITEBOOT
0000 33 environment.
0000 34
0000 35 ENVIRONMENT:
0000 36
0000 37 Called when writing a BOOTBLOCK to an RT-11 device by RTFILREAD.
0000 38
0000 39 ABSTRACT:
0000 40
0000 41 This module provides an interface which allows RTFILREAD to
0000 42 perform in the WRITEBOOT environment. RTFILREAD sometimes
0000 43 runs in a standalone environment in which no VMS system services
0000 44 are available to it. It calls an entrypoint named "FIL$READ_LBN"
0000 45 to read logical blocks for it from the console TU58. This module
0000 46 contains two entrypoints; one which is called prior to the invocation
0000 47 of RTFILREAD and which is passed the string descriptor of the device
0000 48 on which we will write the BOOTBLOCK, the second of which is named
0000 49 "FIL$READ_LBN" and which is called by RTFILREAD.
0000 50
0000 51
0000 52 AUTHOR:
0000 53
0000 54 Robert Rappaport 13 August 1979
0000 55
0000 56 --

```

```

0000 58      .SBTTL RTF$TARGET_DEV - Records the name of Target device.
0000 59
0000 60 :++
0000 61 : Functional description:
0000 62 :
0000 63 : This module provides an interface that allows RTF$OPENFILE to
0000 64 : be used unchanged in the WRITEBOOT environment. When WRITEBOOT
0000 65 : is called upon to write a BOOTBLOCK on an RT-11 structured device
0000 66 : it uses module RTF$OPENFILE to interpret information on the RT-11
0000 67 : device. RTF$OPENFILE was designed to run in a standalone environment
0000 68 : in which no VMS system services are available. RTF$OPENFILE reads
0000 69 : blocks from a TU58 by calling a subroutine called FIL$READ_LBN which
0000 70 : is passed the LBN of the block to be read and a buffer address.
0000 71 :
0000 72 : In the WRITEBOOT environment we have the full VMS environment. So
0000 73 : to read an LBN from an RT-11 device we need only invoke QIO. However
0000 74 : in order to do this we have to know which device we are talking to
0000 75 : so that we can assign it a logical channel. In the standalone
0000 76 : environment the device information is implicit.
0000 77 :
0000 78 : RTF$TARGET_DEV is called from WRITEBOOT prior to the invocation of
0000 79 : RTF$FILEOPEN. The only argument passed is the name of the target
0000 80 : RT-11 device upon which we will later write a BOOTBLOCK. The name
0000 81 : of this device is saved in static storage in variable "TARGET_DEV_NAME"
0000 82 : so that it will be available later when RTF$OPENFILE calls entrypoint
0000 83 : FIL$READ_LBN. At that point we will use the device name to assign
0000 84 : a channel to the device and then issue a QIO to read the block.
0000 85 :
0000 86 :
0000 87 : Inputs:
0000 88 :
0000 89 :     DEVICE_DESC(AP) - string descriptor of target device specification
0000 90 : Outputs:
0000 91 :
0000 92 :     NONE
0000 93 :--
0000 94 :
0000 95 :
0000 96 : Offsets to input arguments:
0000 97 :
0000 98 :
00000004 0000 99      DEVICE_DESC      = 4
0000 100
0000 101
0000 102 : DATA PSECT
0000 103 :
00000000 0000 104      .PSECT  RWDATA,WRT,NOEXE
00000008'00000000 0000 105 TARGET_DEV_DESC:      ; STRING DESCRIPTOR FOR TARGET DEV.
00000008 0008 106      .LONG   0,TARGET_DEV_NAME
00000008 0008 107 TARGET_DEV_NAME:      ; SPACE FOR TARGET DEVICE NAME.
0000001C 0008 108      .BCKB  20
00000000 00000000 001C 109
00000000 00000000 001C 110 IOSB:  .QUAD  0
0000 0024 111 CHAN_WORD:
0000 0024 112      .WORD  0      ; WORD WHEREIN TO STORE CHANNEL NUMBER.
00000000 0026 113
00000000 114      .PSECT  READDEV,EXE,NOWRT

```

```
003C 0000 115 .ENTRY RTF$TARGET_DEV,-
      0002 116 ^M<R2,R3,R4,R5>
      0002 117
      50 04 AC D0 0002 118
00000000'EF 60 B0 0006 119 MOVL DEVICE_DESC(AP),R0 ; R0 => SOURCE DESCRIPTOR.
      000D 120 MOVW (R0),TARGET_DEV_DESC ; COPY LENGTH OF TARGET NAME TO
      000D 121 ; LOCAL DESCRIPTOR.
04 B0 00000000'EF 28 000D 122 MOV3 TARGET_DEV_DESC,@4(R0),TARGET_DEV_NAME
      0015
      001A 123 ; COPY DEVICE NAME.
      001A 124
      04 001A 125 RET
```

```

001B 127      .SBTTL  FIL$READ_LBN - Routine to read a logical block.
001B 128
001B 129 :
001B 130 :++
001B 131 : Functional description:
001B 132 :
001B 133 : This entrypoint is called from RTF$OPENFILE to read a logical
001B 134 : block from the device whose name is defined by the TARGET_DEV_DESC.
001B 135 : This routine simply assigns a channel to this device, issues a QIOW,
001B 136 : and deassigns the channel and returns.
001B 137 :
001B 138 : Inputs:
001B 139 :
001B 140 : LBN(AP) - Logical Block Number of block to be read.
001B 141 :
001B 142 : BUFFER(AP) - Address of the area in memory into which to read block.
001B 143 :
001B 144 :
001B 145 : Outputs:
001B 146 :
001B 147 : R0 - status code of first system service to fail or SSS_NORMAL.
001B 148 :
001B 149 :--
001B 150 :
001B 151 : Offsets to input arguments:
001B 152 :
00000004 001B 153 : LBN = 4
00000008 001B 154 : BUFFER = 8
001B 155 :
001B 156 :
0004 001B 157 : .ENTRY  FIL$READ_LBN,^M<R2>
001D 158 :
001D 159 : $ASSIGN_S      DEVNAM=TARGET_DEV_DESC, -
001D 160 :                CHAN=CHAN_WORD          : CHAN_WORD => ASSIGNED CHANNEL.
52  00000024'EF  E9 0032 161 : BLBC  R0,READ_LBN_RET                : BRANCH AROUND ON FAILURE.
0035 162 : MOVZWL CHAN_WORD,R2                  : COPY CHANNEL # TO R2.
003C 163 :
003C 164 : $QIOW_S      CHAN=R2, -
003C 165 :              FUNC=#IOS_READLBLK, -
003C 166 :              IOSB=IOSB, -
003C 167 :              P1=@BUFFER(AP), -
003C 168 :              P2=#512, -
003C 169 :              P3=LBN(AP)
50  0000001C'EF  E9 0065 170 : BLBC  R0,READ_LBN_RET                : ISSUE QIO AND WAIT.
0068 171 : MOVQ  IOSB,R0                        : BRANCH AROUND ON FAILURE.
006F 172 : BLBC  R0,READ_LBN_RET                : COPY IOSB TO REGISTERS.
0072 173 :
0072 174 : $DASSGN_S    R2                      : DEASSIGN THE CHANNEL.
007C 175 :
007C 176 : READ_LBN_RET:
04  007C 177 : RET
007D 178 : .END
  
```

READLBN
Symbol table

- WRITEBOOT-RTFILREAD I/O interface. ^{L 9}

15-SEP-1984 23:59:00
4-SEP-1984 23:05:21

VAX/VMS Macro V04-00
[BOOTS.SRC]READLBN.MAR;1

Page 5
(3)

```

$ST1      = 00000001
BUFFER    = 00000008
CHAN_WORD = 00000024 R    01
DEVICE_DESC = 00000004
FIL$READ_LBN = 0000001B RG 02
IOS_READ_CBLK ***** X 02
IOSB      = 0000001C R    01
LBN       = 00000004
READ_LBN_RET = 0000007C R  02
RTF$TARGET_DEV = 00000000 RG 02
SYSS$ASSIGN ***** GX 02
SYSS$DASSGN ***** GX 02
SYSS$QIOW ***** GX 02
TARGET_DEV_DESC = 00000000 R 01
TARGET_DEV_NAME = 00000008 R 01

```

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
. ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RWDATA	00000026 (38.)	01 (1.)	NOPIC USR CON REL LCL NOSHR NOEXE RD WRT NOVEC BYTE
READDEV	0000007D (125.)	02 (2.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC BYTE

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
Initialization	30	00:00:00.09	00:00:00.43
Command processing	117	00:00:00.67	00:00:02.73
Pass 1	122	00:00:00.97	00:00:03.83
Symbol table sort	0	00:00:00.01	00:00:00.01
Pass 2	49	00:00:00.39	00:00:00.71
Symbol table output	3	00:00:00.04	00:00:00.04
Psect synopsis output	1	00:00:00.02	00:00:00.03
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	324	00:00:02.19	00:00:07.78

The working set limit was 1050 pages.
3281 bytes (7 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 15 non-local and 0 local symbols.
178 source lines were read in Pass 1, producing 19 object records in Pass 2.
7 pages of virtual memory were used to define 7 macros.

! Macro library statistics !

Macro library name	Macros defined
-----	-----
_\$255\$DUA28:[BOOTS.OBJ]BOOTS.MLB;1	0
-\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
-\$255\$DUA28:[SYSLIB]STARLET.MLB;2	7
TOTALS (all libraries)	7

75 GETS were required to define 7 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:READLBN/OBJ=OBJ\$:READLBN MSRC\$:READLBN/UPDATE=(ENH\$:READLBN)+EXECMLS/LIB+LIB\$:BOOTS.MLB/LIB

