```
FFFFFFFFFFFFFFF         111              111         XXX         XXX
FFFFFFFFFFFFFFF         111              111         XXX         XXX
FFFFFFFFFFFFFFF         111              111         XXX         XXX
FFF                  111111           111111         XXX         XXX
FFF                  111111           111111         XXX         XXX
FFF                  111111           111111         XXX         XXX
FFF                     111              111            XXX   XXX
FFF                     111              111            XXX   XXX
FFF                     111              111            XXX   XXX
FFFFFFFF.FFF            111              111               XXX
FFFFFFFFFFFF            111              111               XXX
FFFFFFFFFFFF            111              111               XXX
FFF                     111              111            XXX   XXX
FFF                     111              111            XXX   XXX
FFF                     111              111            XXX   XXX
FFF                     111              111         XXX         XXX
FFF                     111              111         XXX         XXX
FFF                     111              111         XXX         XXX
FFF                  111111111        111111111      XXX         XXX
FFF                  111111111        111111111      XXX         XXX
FFF                  111111111        111111111      XXX         XXX
```

```
_$25

Symb
----
IOC$
IO_C
IO_C
IO_D
IO_F
IO_S
KILL

KILL
KILL
LB_E
LB_D
LB_F
LB_H
LB_L
LOCA
LOCA
LOCK

LOCK
LOCK
LOCK
LOC_
LOC_
L_CO
L_CO
L_DA
L_DA
MAIN
MAKE
MAKE
MAKE
MAKE
MAKE
MAKE

MAKE
MAKE
MAP_
MAP_

MAP
MARK
MARK
MARK
MARK
MARK
```

```
  CCCCCCCC  HH      HH  KK      KK    SSSSSSSS  UU      UU  MM      MM
  CCCCCCCC  HH      HH  KK      KK    SSSSSSSS  UU      UU  MM      MM
CC          HH      HH  KK      KK  SS          UU      UU  MMMM  MMMM
CC          HH      HH  KK      KK  SS          UU      UU  MMMM  MMMM
CC          HH      HH  KK    KK    SS          UU      UU  MM  MM  MM
CC          HH      HH  KK    KK    SS          UU      UU  MM  MM  MM
CC          HHHHHHHHHH  KKKKK         SSSSSS    UU      UU  MM      MM
CC          HHHHHHHHHH  KKKKK         SSSSSS    UU      UU  MM      MM
CC          HH      HH  KK    KK            SS  UU      UU  MM      MM
CC          HH      HH  KK    KK            SS  UU      UU  MM      MM
CC          HH      HH  KK      KK          SS  UU      UU  MM      MM      ....
CC          HH      HH  KK      KK          SS  UU      UU  MM      MM      ....
  CCCCCCCC  HH      HH  KK      KK  SSSSSSSS    UUUUUUUUUU  MM      MM      ....
  CCCCCCCC  HH      HH  KK      KK  SSSSSSSS    UUUUUUUUUU  MM      MM      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
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

```
0000     1          .TITLE  CHKSUM - Compute checksum routine
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 ;
0000    31 ; FACILITY: F11B XQP
0000    32 ;
0000    33 ; ABSTRACT:
0000    34 ;       This routine computes and checks a file header checksum.
0000    35 ;
0000    36 ;
0000    37 ; ENVIRONMENT: Kernel mode, AST level
0000    38 ;
0000    39 ;
0000    40 ;--
0000    41 ;
0000    42 ; AUTHOR: Christian Saether     , CREATION DATE: 28-Mar-1983
0000    43 ;
0000    44 ; MODIFIED BY:
0000    45 ;
0000    46 ;       V03-001 CDS0001         Christian D. Saether    28-Mar-1983
0000    47 ;               Rewritten in MACRO (from BLISS) to allow optimization
0000    48 ;               of stretching out checksum computation to reduce
0000    49 ;               number of instructions executed and to let various
0000    50 ;               cpu prefetch and pipeline optimizations get somewhere.
0000    51 ;
0000    52
0000    53          .SBTTL  DECLARATIONS
0000    54 ;
0000    55 ; INCLUDE FILES:
0000    56 ;
0000    57
```

```
0000     58 ;
0000     59 ; MACROS:
0000     60 ;
0000     61
0000     62 ;
0000     63 ; EQUATED SYMBOLS:
0000     64 ;
0000     65
0000     66 ;
0000     67 ; OWN STORAGE:
0000     68 ;
0000     69
```

```
                        71          .SBTTL  CHECKSUM
                        72
                        73 ;++
                        74 ;
                        75 ; FUNCTIONAL DESCRIPTION:
                        76 ;
                        77 ;       This routine computes, checks, and stores the file header checksum.
                        78 ;
                        79 ; CALLING SEQUENCE:
                        80 ;       CALLS   #1, CHECKSUM
                        81 ;
                        82 ; INPUT PARAMETERS:
                        83 ;       ARG1:   address of file header buffer
                        84 ;
                        85 ; IMPLICIT INPUTS:
                        86 ;       NONE
                        87 ;
                        88 ; OUTPUT PARAMETERS:
                        89 ;       NONE
                        90 ;
                        91 ; IMPLICIT OUTPUTS:
                        92 ;       NONE
                        93 ;
                        94 ; COMPLETION CODES:
                        95 ;       1 if checksum was correct
                        96 ;       0 if checksum was wrong
                        97 ;
                        98 ; SIDE EFFECTS:
                        99 ;       Correct checksum stored in header
                       100 ;
                       101 ;--
                       102
                       103          .PSECT  $CODE$,NOWRT,EXE,QUAD
                       104
                       105          .ENTRY  CHECKSUM, ^M<R2>
51   D4   0002         106          CLRL    R1                      ; Accumulate checksum in R1.
52 04 AC   D0   0004    107          MOVL    4(AP), R2               ; Header address into R2.
50   20    9A   0008    108          MOVZBL  #32, R0                 ; Loop counter into R0.
     06    11   000B    109          BRB     20$                     ; Only add 7 on first pass.
           01   000D    110          NOP                             ; Quad align 10$ label.
           01   000E    111          NOP
           01   000F    112          NOP
51   82    A0   0010    113 10$:     ADDW2   (R2)+, R1               ; Compute checksum.
51   82    A0   0013    114 20$:     ADDW2   (R2)+, R1               ; Compute checksum.
51   82    A0   0016    115          ADDW2   (R2)+, R1               ; Compute checksum.
51   82    A0   0019    116          ADDW2   (R2)+, R1               ; Compute checksum.
51   82    A0   001C    117          ADDW2   (R2)+, R1               ; Compute checksum.
51   82    A0   001F    118          ADDW2   (R2)+, R1               ; Compute checksum.
51   82    A0   0022    119          ADDW2   (R2)+, R1               ; Compute checksum.
51   82    A0   0025    120          ADDW2   (R2)+, R1               ; Compute checksum.
   E5 50   F5   0028    121          SOBGTR  R0, 10$                 ; 8*32-1 = 255 words summed.
51   62    B1   002B    122          CMPW    (R2), R1                ; Compare with existing checksum.
     02    12   002E    123          BNEQ    30$                     ; Branch if mismatch (leaving R0 = 0).
     50    D6   0030    124          INCL    R0                      ; R0 now set to 1 (success).
62   51    B0   0032    125 30$:     MOVW    R1, (R2)                ; Store computed checksum.
           04   0035    126          RET                             ; Exit.
                0036    127
```

```
                0036    128             .END
```

CHECKSUM 00000000 RG 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
$CODE$	00000036 (54.)	01 (1.)	NOPIC USR CON REL LCL NOSHR EXE RD NOWRT NOVEC QUAD

Performance indicators

Phase	Page faults	CPU Time	Elapsed Time
Initialization	35	00:00:00.10	00:00:00.30
Command processing	135	00:00:00.54	00:00:01.87
Pass 1	71	00:00:00.37	00:00:01.09
Symbol table sort	0	00:00:00.00	00:00:00.00
Pass 2	40	00:00:00.29	00:00:00.66
Symbol table output	2	00:00:00.01	00:00:00.01
Psect synopsis output	1	00:00:00.01	00:00:00.01
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	286	00:00:01.32	00:00:04.13

The working set limit was 750 pages.
1511 bytes (3 pages) of virtual memory were used to buffer the intermediate code.
There were 10 pages of symbol table space allocated to hold 1 non-local and 3 local symbols.
128 source lines were read in Pass 1, producing 13 object records in Pass 2.
0 pages of virtual memory were used to define 0 macros.

Macro library statistics

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

0 GETS were required to define 0 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:CHKSUM/OBJ=OBJ$:CHKSUM MSRC$:CHKSUM/UPDATE=(ENH$:CHKSUM)+EXECML$/LIB

CHKDMO
LIS

CPYNAM
LIS

CLENUP
LIS

CHKPRO
LIS

CHARGEQ
LIS

COMMON
LIS

CREATE
LIS

BADSCN
LIS

CHKHD2
LIS

CHKSUM
LIS

ALLOCB
LIS