ANALYZ

```
000000   BBBBBBBB            JJ  EEEEEEEEEE  XX      XX  EEEEEEEEEE  RRRRRRRR   EEEEEEEEEE   QQQQQQ
000000   BBBBBBBB            JJ  EEEEEEEEEE  XX      XX  EEEEEEEEEE  RRRRRRRR   EEEEEEEEEE   QQQQQQ
00    00 BB      BB          JJ  EE          XX    XX    EE          RR    RR  EE          QQ    QQ
00    00 BB      BB          JJ  EE          XX    XX    EE          RR    RR  EE          QQ    QQ
00    00 BB      BB          JJ  EE            XX  XX    EE          RR    RR  EE          QQ    QQ
00    00 BB      BB          JJ  EE            XX  XX    EE          RR    RR  EE          QQ    QQ
00    00 BBBBBBBB            JJ  EEEEEEE         XX      EEEEEEE     RRRRRRRR   EEEEEEE     QQ    QQ
00    00 BBBBBBBB            JJ  EEEEEEE         XX      EEEEEEE     RRRRRRRR   EEEEEEE     QQ    QQ
00    00 BB      BB  JJ      JJ  EE            XX  XX    EE          RR  RR     EE          QQ    QQ QQ
00    00 BB      BB  JJ      JJ  EE            XX  XX    EE          RR  RR     EE          QQ    QQ QQ
00    00 BB      BB  JJ      JJ  EE          XX    XX    EE          RR    RR   EE          QQ    QQ
00    00 BB      BB  JJ      JJ  EE          XX    XX    EE          RR    RR   EE          QQ    QQ
000000   BBBBBBBB    JJJJJ   EEEEEEEEEE  XX      XX  EEEEEEEEEE  RR      RR  EEEEEEEEEE   QQQQ  QQ
000000   BBBBBBBB    JJJJJ   EEEEEEEEEE  XX      XX  EEEEEEEEEE  RR      RR  EEEEEEEEEE   QQQQ  QQ
```

```
RRRRRRRR   EEEEEEEEEE   QQQQQQ
RRRRRRRR   EEEEEEEEEE   QQQQQQ
RR    RR   EE          QQ    QQ
RR    RR   EE          QQ    QQ
RR    RR   EE          QQ    QQ
RR    RR   EE          QQ    QQ
RRRRRRRR   EEEEEEE     QQ    QQ
RRRRRRRR   EEEEEEE     QQ    QQ
RR  RR     EE          QQ    QQ QQ
RR  RR     EE          QQ    QQ QQ
RR    RR   EE          QQ    QQ
RR    RR   EE          QQ    QQ
RR      RR  EEEEEEEEEE   QQQQ  QQ
RR      RR  EEEEEEEEEE   QQQQ  QQ
```

          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:     VAX/VMS Analyze Command, BLISS Require File
!
! Abstract:     This is the BLISS require file for the ANALYZE facility.
!               It includes various useful constructs and the definitions
!               of all control blocks used by the facility.
!
!
! Environment:
!
! Author: Paul C. Anagnostopoulos, Creation Date: 29 December 1980
!
! Modified By:
!
!     V03-004 DGB0051         Donald G. Blair          10-May-1984
!               Add "severity_level" macro as part of my new
!               condition handling code which allows us
!               to return the correct condition value in R0.
!
!     V03-003 LJA0108         Laurie J. Anderson       30-Jan-1984
!               Add a couple new messages in the list of external literals
!               The message name for bad header block count and indirect
!               message section names.
!
!     V03-002 PCA1011         Paul C. Anagnostopoulos  1-Apr-1983
!               Change the message prefix to ANLRMS$ to ensure that
!               message symbols are unique across all ANALYZEs.  This
!               is necessitated by the new merged message files.
!
!     V03-001 JWT0075         Jim Teague               14-Dec-1982
```

Add message symbol for DMT line.

```
! Here we will define "extensions" to the BLISS language.

! First we need values for boolean variables.

literal
        false           = 0,
        true            = 1;

! Define structure type for VMS structures

structure
        bblock [o,p,s,e;n] =
                [n]
                (bblock+o)<p,s,e>;

! Now we will define macros to generate various things associated with
! string descriptors.

field descriptor_fields = set
        len     = [0,0,16,0],
        ptr     = [4,0,32,0]
tes;

macro descriptor =
        block[8,byte] field(descriptor_fields) %;

macro describe[] =
        uplit long(%charcount(%remaining), uplit byte(%remaining)) %;

macro build_descriptor(name,length,address) =
        (name[0,0,32,0] = length;
        name[ptr] = address)
%;

! Now we define two macros that can generate descripted buffers.  The
! first is for OWN buffers and the second for LOCAL buffers.  Note that
! the local buffer must be defined last in the declarations.

macro own_described_buffer(name,length) =
        name: block[8+length,byte] field(descriptor_fields)
                                initial(length,name+8)
%;

macro lucal_described_buffer(name,length) =
        name: block[8+length,byte] field(descriptor_fields);
        name[0,0,32,0] = length;
        name[ptr] = name+8
%;

! Now we define macros to increment and decrement a variable.

macro increment (var) =
        (var = .var + 1) %,
        decrement (var) =
        (var = .var - 1) %;
```

```
! We need an "infinite" loop contruct.  We also need a more elegant construct
! for terminating a loop.

macro loop =
        while 1 do %;

macro exitif[] =
        if %remaining then exitloop %;

! Define a macro that can check statuses from routines.

macro check(status)[] =
        (if not status then
                signal(%remaining);)
%;

! Macro to implement a function (f) of the message severity level that
! maps the various severity levels such that arithmetic comparisions of the
! mapped values (  f(severity) )  yield a order of precedence that is
! intuivitvely acceptable:
!
!               ERROR NAME       OLDVAL        NEWVAL
!
!               F(SUCCESS)         1    -->      0
!               F(INFORMATIONAL)   3    -->      2
!               F(WARNING)         0    -->      3
!               F(ERROR)           2    -->      5
!               F(SEVERE_ERROR)    4    -->      7

macro
severity_level (tmp_status) =
        BEGIN
        LOCAL tmp_code: BBLOCK [LONG];
        tmp_code = tmp_status;
        .tmp_code [sts$v_severity] - (4 * .tmp_code [sts$v_success]) + 3
        END%;

! Define literals for useful control characters.

literal
        bell            = %x'07',
        backspace       = %x'08',
        tab             = %x'09',
        linefeed        = %x'0a',
        formfeed        = %x'0c',
        creturn         = %x'0d',
        ctrl_u          = %x'15',
        ctrl_w          = %x'17',
        ctrl_z          = %x'1a',
        escape          = %x'1b',
        delete          = %x'7f';
```

```
! The following macros allow us to deal with fields in variable-length
! records.  Since we are analyzing, we can never assume that a field exists.

! This first macro requires five arguments:
!       1-4)    A structure reference to a field in a block.
!       5)      The address of a descriptor of the record containing the block.
! Upon entry, we assume two things:
!               FIT_OK says whether all fields have fit so far.
!               SCANP points to the block within the record.
! What we will do is ensure that the specified field, defined relative to
! SCANP, fits within the specified record.  If not, we will produce an
! error messages and clear FIT_OK.

macro ensure_field_fit(position,offset,size,extension,record_dsc) =
        if .fit_ok then
                if .scanp+position+size/8 gtru .record_dsc[ptr]+.record_dsc[len] then (
                        anl$format_error(anlobj$_fieldfit);
                        fit_ok = false;
                );
%;

! The next macro requires exactly the same five arguments, and makes the
! same two assumptions.  However, in this case, we assume the field describes
! the count byte of an ASCIC string.  We will check the fit of both the
! count byte and the string itself.  We will also construct a descriptor
! of the string in the sixth argument for later use.

macro ensure_ascic_fit(position,offset,size,extension,record_dsc,ascic_dsc) =
        if .fit_ok then (
                ensure_field_fit(position,offset,size,extension,record_dsc);
                if .fit_ok then (
                        build_descriptor(ascic_dsc,ch$rchar(.scanp+position),.scanp+position+1);
                        ensure_field_fit(offset+1,0,.ascic_dsc[len],0,record_dsc);
                );
        );
%;
```

! Include the definitions of all the ridiculous message status codes.

external literal
        anlobj$_ok,
        anlobj$_anything,
        anlobj$_datatype,
        anlobj$_errorcount,
        anlobj$_errornone,
        anlobj$_errors,
        anlobj$_exefixa,
        anlobj$_exefixaimage,
        anlobj$_exefixaline,
        anlobj$_exefixcount,
        anlobj$_exefixextra,
        anlobj$_exefixfixed,
        anlobj$_exefixflags,
        anlobj$_exefixg,
        anlobj$_exefixgimage,
        anlobj$_exefixgline,
        anlobj$_exefixlist,
        anlobj$_exefixname,
        anlobj$_exefixname0,
        anlobj$_exefixp,
        anlobj$_exefixpsect,
        anlobj$_exefixup,
        anlobj$_exefixupnone,
        anlobj$_exegst,
        anlobj$_exehdr,
        anlobj$_exehdractive,
        anlobj$_exehdrblkcount,
        anlobj$_exehdrchancount,
        anlobj$_exehdrchandef,
        anlobj$_exehdrdececo,
        anlobj$_exehdrdmt,
        anlobj$_exehdrdst,
        anlobj$_exehdrfileid,
        anlobj$_exehdrfixed,
        anlobj$_exehdrflags,
        anlobj$_exehdrgblident,
        anlobj$_exehdrgst,
        anlobj$_exehdrident,
        anlobj$_exehdrimageid,
        anlobj$_exehdrisd,
        anlobj$_exehdrisdbase,
        anlobj$_exehdrisdcount,
        anlobj$_exehdrisdflags,
        anlobj$_exehdrisdgblnam,
        anlobj$_exehdrisdnum,
        anlobj$_exehdrisdpfcdef,
        anlobj$_exehdrisdpfcsiz,
        anlobj$_exehdrisdtype,
        anlobj$_exehdrisdvbn,
        anlobj$_exehdrlinkid,
        anlobj$_exehdrmatch,
        anlobj$_exehdrname,
        anlobj$_exehdrnopatch,

```
anlobj$_exehdrpagecount,
anlobj$_exehdrpagedef,
anlobj$_exehdrpatch,
anlobj$_exehdrpatchdate,
anlobj$_exehdrpriv,
anlobj$_exehdrropatch,
anlobj$_exehdrrwpatch,
anlobj$_exehdrsymdbg,
anlobj$_exehdrsysver,
anlobj$_exehdrtextvbn,
anlobj$_exehdrtime,
anlobj$_exehdrtypeexe,
anlobj$_exehdrtypelim,
anlobj$_exehdrusereco,
anlobj$_exehdrxfer1,
anlobj$_exehdrxfer2,
anlobj$_exehdrxfer3,
anlobj$_exeheading,
anlobj$_exepatch,
anlobj$_flag,
anlobj$_hexdata,
anlobj$_hexheading1,
anlobj$_hexheading2,
anlobj$_indmsgsec,
anlobj$_interact,
anlobj$_mask,
anlobj$_objcprrec,
anlobj$_objdbgrec,
anlobj$_objenv,
anlobj$_objeomflags,
anlobj$_objeomrec,
anlobj$_objeomsevabt,
anlobj$_objeomseverr,
anlobj$_objeomsevign,
anlobj$_objeomsevres,
anlobj$_objeomsevsuc,
anlobj$_objeomsevwrn,
anlobj$_objeomwrec,
anlobj$_objfadpassmech,
anlobj$_objgsdenv,
anlobj$_objgsdenvflags,
anlobj$_objgsdenvpar,
anlobj$_objgsdepm,
anlobj$_objgsdepmw,
anlobj$_objgsdidc,
anlobj$_objgsdidcent,
anlobj$_objgsdidcflags,
anlobj$_objgsdidcmatch,
anlobj$_objgsdidcobj,
anlobj$_objgsdidcvala,
anlobj$_objgsdidcvalb,
anlobj$_objgsdlepm,
anlobj$_objgsdlpro,
anlobj$_objgsdlsy,
anlobj$_objgsdpro,
anlobj$_objgsdprow,
```

```
anlobj$_objgsdpsc,
anlobj$_objgsdpscalign,
anlobj$_objgsdpscalloc,
anlobj$_objgsdpscbase,
anlobj$_objgsdpscflags,
anlobj$_objgsdrec,
anlobj$_objgsdspsc,
anlobj$_objgsdsym,
anlobj$_objgsdsymw,
anlobj$_objgtxrec,
anlobj$_objhdrignrec,
anlobj$_objheading,
anlobj$_objlitindex,
anlobj$_objlnkrec,
anlobj$_objlnmrec,
anlobj$_objmhdcreate,
anlobj$_objmhdname,
anlobj$_objmhdpatch,
anlobj$_objmhdrec,
anlobj$_objmhdrecsiz,
anlobj$_objmhdstrlvl,
anlobj$_objmhdversion,
anlobj$_objmtccorrect,
anlobj$_objmtcinput,
anlobj$_objmtcname,
anlobj$_objmtcrec,
anlobj$_objmtcseqnum,
anlobj$_objmtcuic,
anlobj$_objmtcversion,
anlobj$_objmtcwhen,
anlobj$_objproargcount,
anlobj$_objproargnum,
anlobj$_objpsect,
anlobj$_objsrcrec,
anlobj$_objstatheading1,
anlobj$_objstatheading2,
anlobj$_objstatline,
anlobj$_objstattotal,
anlobj$_objsymbol,
anlobj$_objsymflags,
anlobj$_objtirargindex,
anlobj$_objtircmd,
anlobj$_objtircmdstk,
anlobj$_objtbtrec,
anlobj$_objtirrec,
anlobj$_objtirstoim,
anlobj$_objtirvield,
anlobj$_objttlrec,
anlobj$_objvalue,
anlobj$_objuvalue,
anlobj$_protection,
anlobj$_severity,
anlobj$_text,
anlobj$_texthdr,
anlobj$_nosuchmod,
anlobj$_baddate,
```

```
        anlobj$_badhdrblkcount,
        anlobj$_badseverity,
        anlobj$_badsymlst,
        anlobj$_badsymchar,
        anlobj$_badsymlen,
        anlobj$_exebadfixupend,
        anlobj$_exebadfixupisd,
        anlobj$_exebadfixupvbn,
        anlobj$_exebadisds1,
        anlobj$_exebadisdtype,
        anlobj$_exebadmatch,
        anlobj$_exebadpatchlen,
        anlobj$_exebadobj,
        anlobj$_exebadtype,
        anlobj$_exebadxfer0,
        anlobj$_exehdrisdlong,
        anlobj$_exehdrlong,
        anlobj$_exeisdlendzro,
        anlobj$_exeisdlengbl,
        anlobj$_exeisdlenpriv,
        anlobj$_exenotnative,
        anlobj$_extrabytes,
        anlobj$_fieldfit,
        anlobj$_flagerror,
        anlobj$_notok,
        anlobj$_objbadidcmatch,
        anlobj$_objbadnum,
        anlobj$_objbadpop,
        anlobj$_objbadpush,
        anlobj$_objbadtype,
        anlobj$_objbadvield,
        anlobj$_objeombadsev,
        anlobj$_objeommissing,
        anlobj$_objfadbadavc,
        anlobj$_objfadbadrbc,
        anlobj$_objgsdbadalign,
        anlobj$_objgsdbadsubtyp,
        anlobj$_objhdrres,
        anlobj$_objmhdbadrecsiz,
        anlobj$_objmhdbadstrlvl,
        anlobj$_objmhdmissing,
        anlobj$_objnontircmd,
        anlobj$_objnopsc,
        anlobj$_objnullrec,
        anlobj$_objp0space,
        anlobj$_objprominmax,
        anlobj$_objpscabslen,
        anlobj$_objrectoobig,
        anlobj$_objtirres,
        anlobj$_objundefenv,
        anlobj$_objundeflit,
        anlobj$_objundefpsc,
        analyze$_facility;

! We use a few of the message in the shareable message file SHRMSG.
! Define status codes for these which include our facility code and
```

! the message severity.

literal
        anlobj$_closein = shr$_closein + 177*65536 + sts$k_error,
        anlobj$_closeout   = shr$_closeout + 177*65536 + sts$k_error,
        anlobj$_openin = shr$_openin + 177*65536 + sts$k_error,
        anlobj$_openout = shr$_openout + 177*65536 + sts$k_severe,
        anlobj$_readerr = shr$_readerr + 177*65536 + sts$k_error,
        anlobj$_writeerr   = shr$_writeerr + 177*65536 + sts$k_severe;