U.S. DEPARTMENT OF THE INTERIOR
U.S. GEOLOGICAL SURVEY

MAKESHARE.COM

A VMS Utility for Creating Shareable Images from Object Module Libraries

Lawrence M. Baker

OPEN-FILE REPORT 91-379

This report is preliminary and has not been reviewed for conformity with U.S. Geological Survey editorial standards. Any use of trade, product or firm names is for descriptive purposes only and does not imply endorsement by the U.S. Government. Although this program has been used by the U.S. Geological Survey, no warranty, expressed or implied, is made by the USGS as to the accuracy and functioning of the program and related program material, nor shall the fact of distribution constitute any such warranty, and no responsibility is assumed by the USGS in connection therewith.

Menlo Park, California

1991

July 1991

Open File Report 91-379

MAKESHARE.COM

A VMS Utility for Creating Shareable Images from Object Module Libraries

1 INTRODUCTION

Shareable images offer VMS users significant advantages as replacements for conventional object module libraries. Disk space is conserved because a single disk copy of a shareable image is incorporated by reference into every program which uses it. Time is saved because the Linker collects the necessary object modules and resolves their external references when the shareable image is created, and image activation is quicker, especially if the shareable image is installed. Less memory is used because a single memory copy of a shareable image is shared by all running programs which reference it (provided the system manager has installed it).

Shareable images are no more difficult to use than object module libraries, since they can be inserted into *shareable image* libraries, which can be used with the Linker anywhere an object module library can be used. (Actually, the shareable image itself is not inserted into the library, just the entry point names. The shareable image must still be maintained as a separate file, usually in the same directory with its library.)

Shareable images encourage good software engineering practices. A shareable image exports only those entry points explicitly declared *Universal.* Entry points that are not declared Universal — for example, those that are intended for use only within the shareable image — are not made available by the Linker when the shareable image is incorporated into a program. Read-only data areas in the shareable image can be shared along with the code by all referencing programs. Read/write data areas can be either shared among all referencing programs, or a private copy can be automatically created for each program when it is activated.

Shareable images ease program maintenance. An entry point in a shareable image can be called indirectly through its corresponding entry in a special jump table, called a *transfer vector* (normally placed at the bottom of the image). If the size and location of the transfer vector table is fixed, the shareable image can be updated without requiring referencing programs to re-link with the new version — to fix bugs, for example. In fact, so long as the existing transfer vectors are preserved in the new version at their same locations, additional transfer vectors can be added to the shareable image to support new features without disturbing programs which have been linked to the old version.

When a program linked to a shareable image is run, the image activator locates the shareable image file by translating the shareable image name as a logical name. Thus, users can easily choose between multiple versions of a shareable image — a production version and a test version, perhaps — by simply changing the definition of a logical name.

While the creation of a shareable image is reasonably straightforward, it can be tedious. This is where MakeShare can help. If you have an object module library which you would like to bind into a shareable image, MakeShare will create a default shareable image and its library. In addition, MakeShare will create all the files necessary to build a customized version of the shareable image and its library, complete with transfer vectors.

The Linker manual [6] contains a far more detailed discussion of shareable images, including the commands necessary to create them, install them, and use them. It may be useful to keep it nearby for reference while reading the rest of this report.

2 COMMAND LINE

MakeShare is a DCL command procedure, which is invoked by prefixing "@" to the file specification for `MakeShare.com`, followed by any arguments. Alternately, the system manager may define `MakeShare` as a DCL global symbol which equates to `@MakeShare.com`. In that case, MakeShare can be invoked as a DCL command.

In the command line format given below and in the examples which follow, it is assumed that `MakeShare.com` is a file in the user's default directory. If `MakeShare` has been defined as a DCL global symbol, omit the "@" prefix.

Format

`@MakeShare` library_name [shareable_image_name]

Parameters

library_name

Specifies the name of the object module library to be bound, without the trailing `.olb` extension.

library_name must be specified; there is no default.

shareable_image_name

Specifies the name to be given to the shareable image and its library, also without any trailing file type extension. The file name part of shareable_image_name is limited to 23 characters.

If shareable_image_name is not supplied, it is constructed by adding an "`Shr`" suffix to library_name.

While both the library_name and the shareable_image_name may include any part of a fully qualified file specification — except for the file type and version number — it is recommended that they consist only of the file name part. This simplifies the contents of the command files and source files that will be created; it allows the files to be migrated to another location in the file system (so, for example, you can maintain a test version and a production version in different directories); and it avoids generating symbols and command lines which exceed DCL's limits.

3 OPERATION

3.1 Input and Output Files

MakeShare requires a single input file: the object module library to be bound into a shareable image (library_name.olb). MakeShare creates eight output files, which all share the same file name part (shareable_image_name) and differ only in the file type extension:

`.lis` Librarian listing of the object module library.

`.exe` Default shareable image. Every entry point is declared Universal and all Fortran common blocks are left shareable.

`.map` Linker map of the default shareable image.

`.olb` Shareable image library for the default shareable image.

`.bld` DCL command procedure containing the necessary commands to build a version of the shareable image with every entry point declared Universal, and with all Fortran common blocks marked non-shareable (between programs, not within a program). It also contains commands (commented out) to build a version using transfer vectors.

`.opt` Linker options file used by the `.bld` file. It contains options (commented out) to build a version using transfer vectors and instructions for modifying the file to make use of them.

`.vec` Macro-32 source file containing transfer vectors for all the entry points in the object module library. Any entry points which are for internal use only should be commented out. In addition, the `GSMatch` criteria in the Linker options file must be compatible with the `.Ident` in this file. To maintain upward compatibility, if future versions of the shareable image add new entry points, their transfer vectors should be added to the end of the file only, and `.Ident` should be updated accordingly.

`.Release_Notes` Release notes file containing instructions for customizing the shareable image and installing it.

If MakeShare terminates prematurely, you may also find a temporary `.tmp` file containing the output from a search of the `.map` file, which should be deleted.

3.2 Method of Operation

MakeShare validates its input arguments by attempting to locate the file library_name`.olb` and by checking the syntax and length of shareable_image_name. It assumes the user has permission to create files in the directory specified in or implied by shareable_image_name.

Then MakeShare creates a Librarian listing of the object module library. This listing contains the names of all the modules in the library and their respective entry points. MakeShare creates `/Include` commands in a Linker options file for every module name in the Librarian listing file. In addition, it creates `Universal` commands in the Linker options file and a `Transfer_Vector` entry in a Macro-32 transfer vectors source file for every entry point name in the Librarian listing file. MakeShare also creates a DCL command procedure file at this time for creating the shareable image and its library.

Using the DCL command procedure file and the Linker options file just created, MakeShare then creates the default shareable image and its library, along with a Linker map file. The map file is searched to locate any program sections (PSects) which look like Fortran common blocks, *i.e.*, with attributes "OVR,REL,GBL,SHR,NOEXE,RD,WRT." For each Fortran common block found, MakeShare appends a PSect_Attr command to the Linker options file which changes its shareability attribute from SHR (shareable across all referencing programs) to NOSHR (shareable only within a program).

The release notes file is then created, and MakeShare exits.

As it is running, MakeShare prints informational messages (listed in Section 5.1) to indicate its progress.

4 EXAMPLE

The following example will walk the reader through the creation of a shareable image and library for object module library DistribLib.olb.

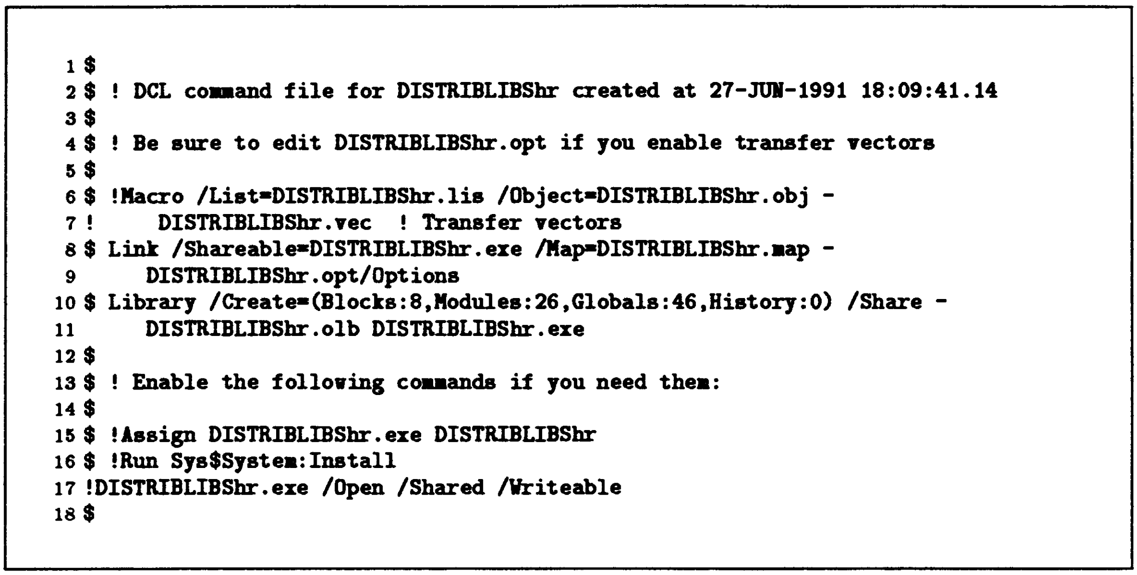
Step 1: Run MakeShare.com

To run MakeShare on DistribLib.olb, taking the default for the name of the shareable image (DistribLibShr), type:

```
$ @MakeShare DistribLib

%MAKESHARE-I-START, Make of DISTRIBLIBShr started at 27-JUN-1991 18:09:23.13.

%MAKESHARE-I-LIBLIST, Creating Librarian listing file . . .
%MAKESHARE-I-OPTIONS, Creating Linker options file and Macro-32 transfer vectors
 source file . . .
%MAKESHARE-I-DCLPROC, Creating DCL command procedure file . . .
%MAKESHARE-I-LINK, Creating default shareable image . . .
%MAKESHARE-I-OPTIONS, Appending Linker options for Fortran R/W COMMONs . . .
%MAKESHARE-I-NOTES, Creating DISTRIBLIBShr.Release_Notes . . .

Read the release notes file DISTRIBLIBShr.Release_Notes to customize and install
 DISTRIBLIBShr.

%MAKESHARE-I-COMPLETED, Make of DISTRIBLIBShr completed at 27-JUN-1991 18:10:09.
64.
```

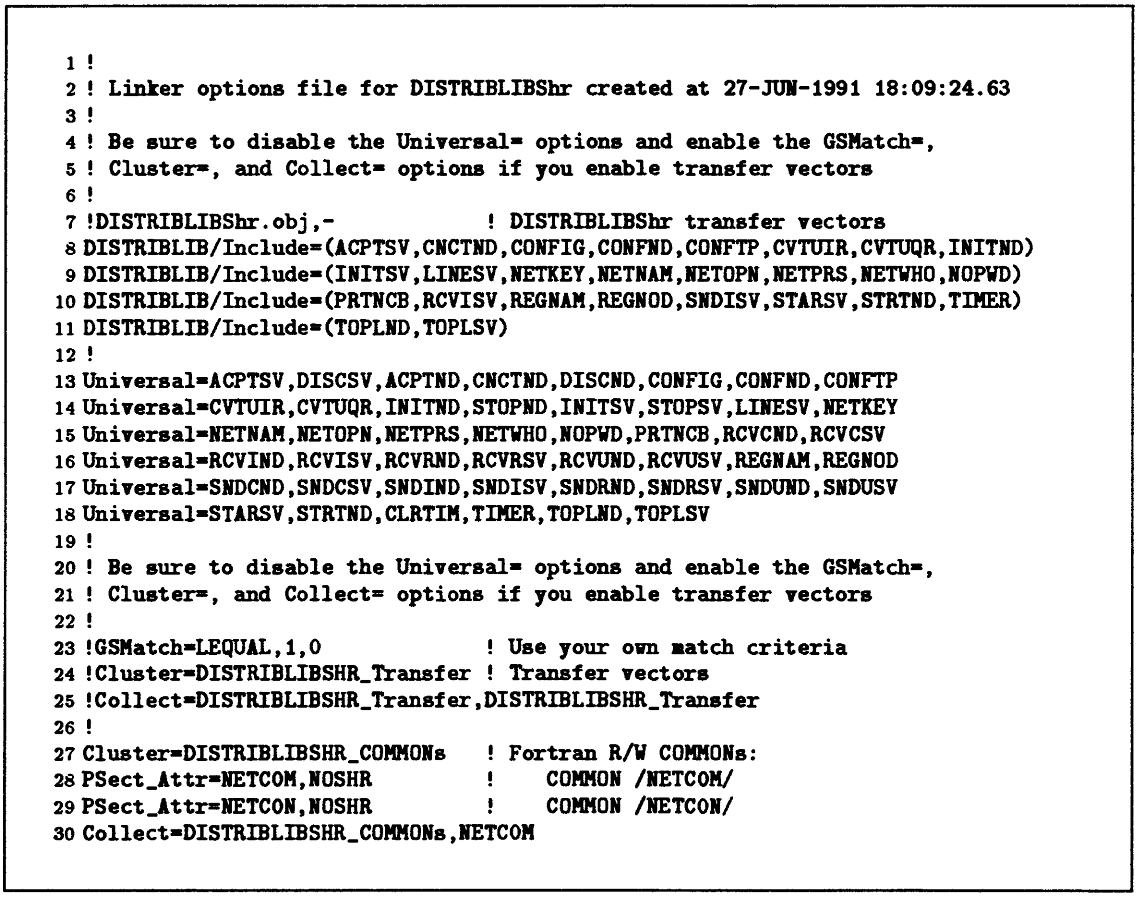
To see that the eight new files are there, along with the original object module library, type:

```
$ Directory DistribLib*

Directory PUB1:[BAKER]

DISTRIBLIB.OLB;1    DISTRIBLIBSHR.BLD;1 DISTRIBLIBSHR.EXE;1 DISTRIBLIBSHR.LIS;1
DISTRIBLIBSHR.MAP;1 DISTRIBLIBSHR.OLB;1 DISTRIBLIBSHR.OPT;1
DISTRIBLIBSHR.RELEASE_NOTES;1           DISTRIBLIBSHR.VEC;1

Total of 9 files.
```



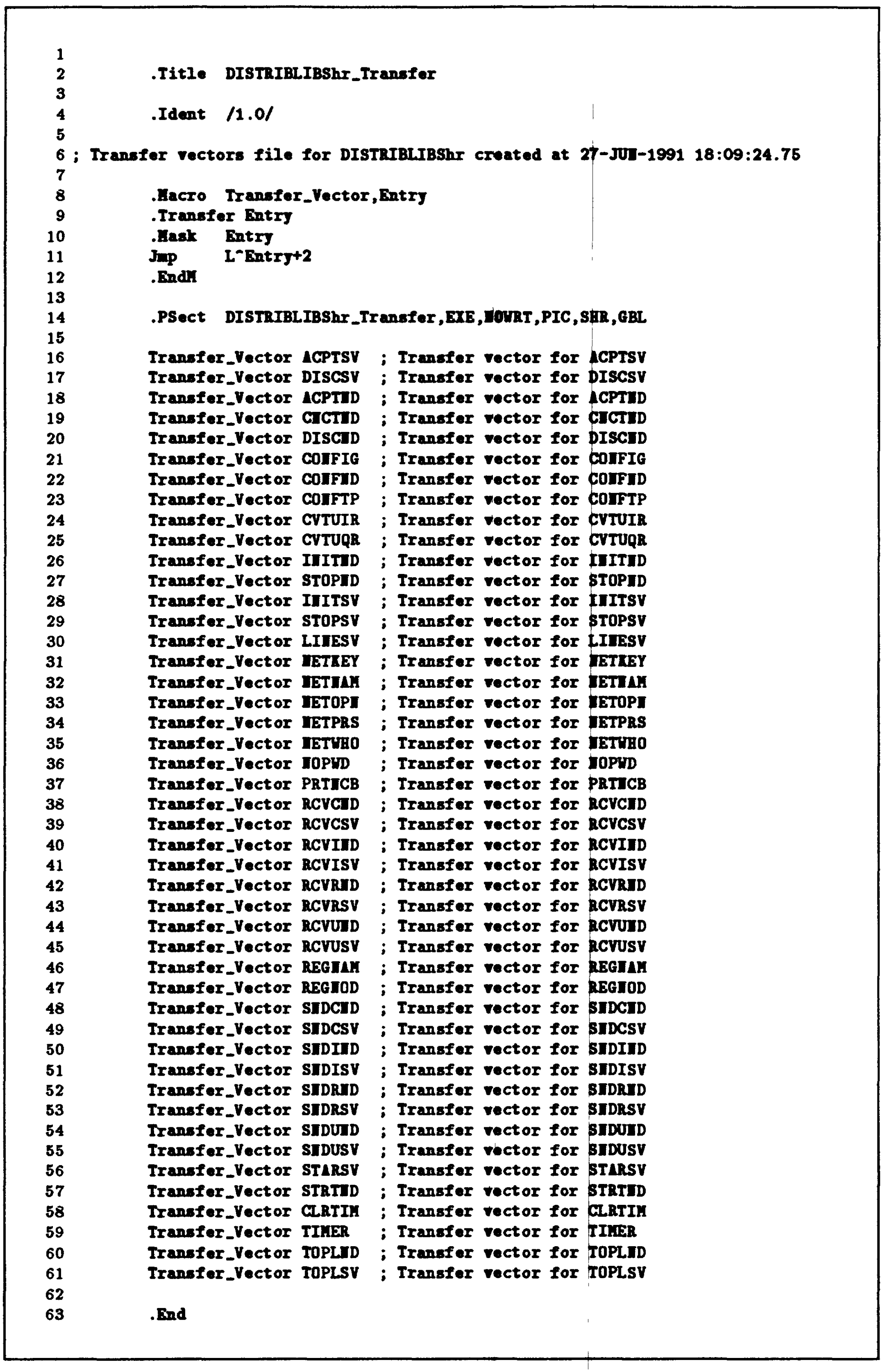
```
$
$ ! DCL command file for DISTRIBLIBShr created at 27-JUN-1991 18:09:41.14
$
$ ! Be sure to edit DISTRIBLIBShr.opt if you enable transfer vectors
$
$ !Macro /List=DISTRIBLIBShr.lis /Object=DISTRIBLIBShr.obj -
!      DISTRIBLIBShr.vec  ! Transfer vectors
$ Link /Shareable=DISTRIBLIBShr.exe /Map=DISTRIBLIBShr.map -
      DISTRIBLIBShr.opt/Options
$ Library /Create=(Blocks:8,Modules:26,Globals:46,History:0) /Share -
      DISTRIBLIBShr.olb DISTRIBLIBShr.exe
$
$ ! Enable the following commands if you need them:
$
$ !Assign DISTRIBLIBShr.exe DISTRIBLIBShr
$ !Run Sys$System:Install
!DISTRIBLIBShr.exe /Open /Shared /Writeable
$
```

Figure 1. Original DCL command procedure file (DistribLibShr.bld;1).



```
!
! Linker options file for DISTRIBLIBShr created at 27-JUN-1991 18:09:24.63
!
! Be sure to disable the Universal= options and enable the GSMatch=,
! Cluster=, and Collect= options if you enable transfer vectors
!
!DISTRIBLIBShr.obj,-              ! DISTRIBLIBShr transfer vectors
DISTRIBLIB/Include=(ACPTSV,CNCTND,CONFIG,CONFND,CONFTP,CVTUIR,CVTUQR,INITND)
DISTRIBLIB/Include=(INITSV,LINESV,NETKEY,NETNAM,NETOPN,NETPRS,NETWHO,NOPWD)
DISTRIBLIB/Include=(PRTNCB,RCVISV,REGNAM,REGNOD,SNDISV,STARSV,STRTND,TIMER)
DISTRIBLIB/Include=(TOPLND,TOPLSV)
!
Universal=ACPTSV,DISCSV,ACPTND,CNCTND,DISCND,CONFIG,CONFND,CONFTP
Universal=CVTUIR,CVTUQR,INITND,STOPND,INITSV,STOPSV,LINESV,NETKEY
Universal=NETNAM,NETOPN,NETPRS,NETWHO,NOPWD,PRTNCB,RCVCND,RCVCSV
Universal=RCVIND,RCVISV,RCVRND,RCVRSV,RCVUND,RCVUSV,REGNAM,REGNOD
Universal=SNDCND,SNDCSV,SNDIND,SNDISV,SNDRND,SNDRSV,SNDUND,SNDUSV
Universal=STARSV,STRTND,CLRTIM,TIMER,TOPLND,TOPLSV
!
! Be sure to disable the Universal= options and enable the GSMatch=,
! Cluster=, and Collect= options if you enable transfer vectors
!
!GSMatch=LEQUAL,1,0               ! Use your own match criteria
!Cluster=DISTRIBLIBSHR_Transfer ! Transfer vectors
!Collect=DISTRIBLIBSHR_Transfer,DISTRIBLIBSHR_Transfer
!
Cluster=DISTRIBLIBSHR_COMMONs    ! Fortran R/W COMMONs:
PSect_Attr=NETCOM,NOSHR          !    COMMON /NETCOM/
PSect_Attr=NETCON,NOSHR          !    COMMON /NETCON/
Collect=DISTRIBLIBSHR_COMMONs,NETCOM
```

Figure 2. Original Linker options file (DistribLibShr.opt;1).



```

        .Title  DISTRIBLIBShr_Transfer

        .Ident  /1.0/

; Transfer vectors file for DISTRIBLIBShr created at 27-JUN-1991 18:09:24.75

        .Macro  Transfer_Vector,Entry
        .Transfer Entry
        .Mask   Entry
        Jmp     L^Entry+2
        .EndM

        .PSect  DISTRIBLIBShr_Transfer,EXE,NOWRT,PIC,SHR,GBL

        Transfer_Vector ACPTSV  ; Transfer vector for ACPTSV
        Transfer_Vector DISCSV  ; Transfer vector for DISCSV
        Transfer_Vector ACPTND  ; Transfer vector for ACPTND
        Transfer_Vector CNCTND  ; Transfer vector for CNCTND
        Transfer_Vector DISCND  ; Transfer vector for DISCND
        Transfer_Vector CONFIG  ; Transfer vector for CONFIG
        Transfer_Vector CONFND  ; Transfer vector for CONFND
        Transfer_Vector CONFTP  ; Transfer vector for CONFTP
        Transfer_Vector CVTUIR  ; Transfer vector for CVTUIR
        Transfer_Vector CVTUQR  ; Transfer vector for CVTUQR
        Transfer_Vector INITND  ; Transfer vector for INITND
        Transfer_Vector STOPND  ; Transfer vector for STOPND
        Transfer_Vector INITSV  ; Transfer vector for INITSV
        Transfer_Vector STOPSV  ; Transfer vector for STOPSV
        Transfer_Vector LINESV  ; Transfer vector for LINESV
        Transfer_Vector NETKEY  ; Transfer vector for NETKEY
        Transfer_Vector NETNAM  ; Transfer vector for NETNAM
        Transfer_Vector NETOPN  ; Transfer vector for NETOPN
        Transfer_Vector NETPRS  ; Transfer vector for NETPRS
        Transfer_Vector NETWHO  ; Transfer vector for NETWHO
        Transfer_Vector NOPWD   ; Transfer vector for NOPWD
        Transfer_Vector PRTNCB  ; Transfer vector for PRTNCB
        Transfer_Vector RCVCND  ; Transfer vector for RCVCND
        Transfer_Vector RCVCSV  ; Transfer vector for RCVCSV
        Transfer_Vector RCVIND  ; Transfer vector for RCVIND
        Transfer_Vector RCVISV  ; Transfer vector for RCVISV
        Transfer_Vector RCVRND  ; Transfer vector for RCVRND
        Transfer_Vector RCVRSV  ; Transfer vector for RCVRSV
        Transfer_Vector RCVUND  ; Transfer vector for RCVUND
        Transfer_Vector RCVUSV  ; Transfer vector for RCVUSV
        Transfer_Vector REGNAM  ; Transfer vector for REGNAM
        Transfer_Vector REGNOD  ; Transfer vector for REGNOD
        Transfer_Vector SNDCND  ; Transfer vector for SNDCND
        Transfer_Vector SNDCSV  ; Transfer vector for SNDCSV
        Transfer_Vector SNDIND  ; Transfer vector for SNDIND
        Transfer_Vector SNDISV  ; Transfer vector for SNDISV
        Transfer_Vector SNDRND  ; Transfer vector for SNDRND
        Transfer_Vector SNDRSV  ; Transfer vector for SNDRSV
        Transfer_Vector SNDUND  ; Transfer vector for SNDUND
        Transfer_Vector SNDUSV  ; Transfer vector for SNDUSV
        Transfer_Vector STARSV  ; Transfer vector for STARSV
        Transfer_Vector STRTND  ; Transfer vector for STRTND
        Transfer_Vector CLRTIM  ; Transfer vector for CLRTIM
        Transfer_Vector TIMER   ; Transfer vector for TIMER
        Transfer_Vector TOPLND  ; Transfer vector for TOPLND
        Transfer_Vector TOPLSV  ; Transfer vector for TOPLSV

        .End
```

Figure 3. Original Macro-32 transfer vectors source file (`DistribLibShr.vec;1`).

```

                              RELEASE NOTES

DISTRIBLIBShr.exe has been built and inserted into DISTRIBLIBShr.olb.  Before it
can be installed, all non-shared data areas, e.g., Fortran read/write COMMONs,
must be given the NOSHR attribute.  DISTRIBLIBShr.opt has been created for you
with all the Linker options necessary to build a new version of
DISTRIBLIBShr.exe with all COMMONs marked NOSHR.  You may edit DISTRIBLIBShr.opt
to remove the PSect_Attr option for any COMMONs that are to be shared, i.e.,
that contain only read-only data, and to properly set the GSMATCH option.  If
you want to use transfer vectors in the new image, edit DISTRIBLIBShr.bld and
DISTRIBLIBShr.opt, and possibly the .Ident in DISTRIBLIBShr.vec (to match the
GSMatch option).  To build a new version of DISTRIBLIBShr.exe, type

     $ @DISTRIBLIBShr.bld

You must define the logical name DISTRIBLIBShr to refer to the image file if you
do not move DISTRIBLIBShr.exe to Sys$Share:.  Also, you must install the image
to enable sharing of physical memory.  (Use /Writeable only if shared read-write
access is required.)  Sample commands are included (but commented out) in
DISTRIBLIBShr.bld.

For additional information on creating and using shareable images, see the VMS
Linker Utility Manual, Chapter 4.

```

Figure 4. Release notes file (`DistribLibShr.Release_Notes;1`).

Figures 1 to 3 show the DCL command procedure file, the Linker options file, and the Macro-32 transfer vectors source file produced by MakeShare. To use transfer vectors, these files must be edited. Figure 4 shows the release notes file.

Step 2: Edit Files

The DCL command procedure file (Figure 1) already has the necessary `Macro` command to assemble the Macro-32 transfer vectors source file commented out in lines 6 and 7. To enable them, remove the "`!`" comment character at the front of these lines. The resulting DCL command procedure file is shown in Figure 5.

The Linker options file (Figure 2) must be modified to remove the "`!`" comment character at the front of line 7 to include the transfer vectors object file in the shareable image. The `Universal` commands (lines 13 through 18) must then be removed, since transfer vectors are automatically declared Universal by the Linker. Lines 23 through 25 must also be enabled to establish the `GSMatch` criteria and collect the transfer vectors object module into its own image section. Finally, the list of Fortran common blocks at the end of the file must be inspected to determine if any of them should be removed from the list (normally none). The resulting Linker options file is shown in Figure 6.

The last file which must be modified is the Macro-32 transfer vectors source file (Figure 3). Many of the entry points in the object module library are for internal use only. Also, it is preferable that the remaining entry points appear in the same order in the Macro-32

```
$
$ ! DCL command file for DISTRIBLIBShr created at 27-JUN-1991 18:09:41.14
$
$ ! Be sure to edit DISTRIBLIBShr.opt if you enable transfer vectors
$
$ Macro /List=DISTRIBLIBShr.lis /Object=DISTRIBLIBShr.obj -
      DISTRIBLIBShr.vec  ! Transfer vectors
$ Link /Shareable=DISTRIBLIBShr.exe /Map=DISTRIBLIBShr.map -
      DISTRIBLIBShr.opt/Options
$ Library /Create=(Blocks:8,Modules:26,Globals:46,History:0) /Share -
      DISTRIBLIBShr.olb DISTRIBLIBShr.exe
$
$ ! Enable the following commands if you need them:
$
$ !Assign DISTRIBLIBShr.exe DISTRIBLIBShr
$ !Run Sys$System:Install
!DISTRIBLIBShr.exe /Open /Shared /Writeable
$
```

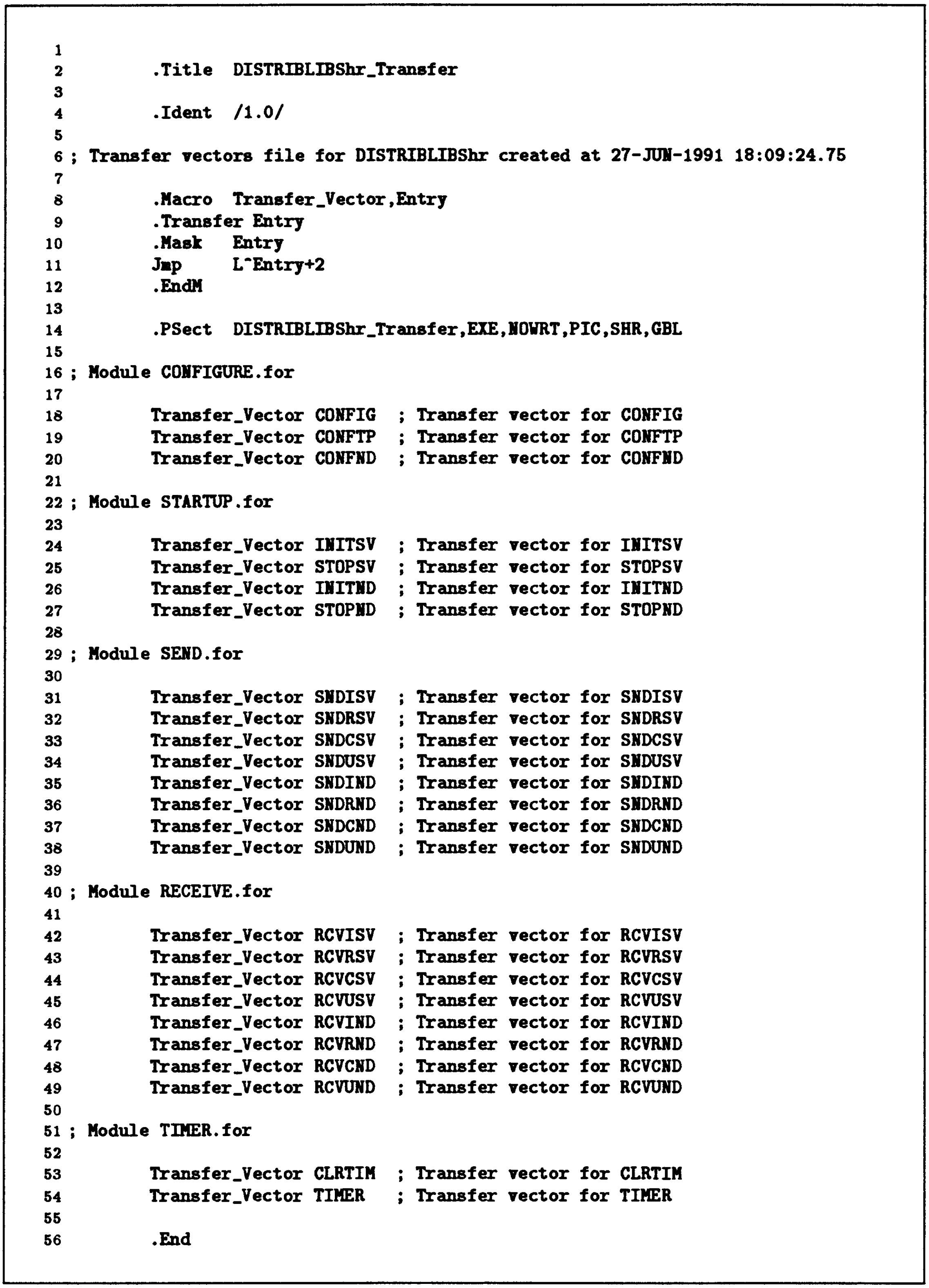
Figure 5. Modified DCL command procedure file (`DistribLibShr.bld;2`).

```
!
! Linker options file for DISTRIBLIBShr created at 27-JUN-1991 18:09:24.63
!
DISTRIBLIBShr.obj,-               ! DISTRIBLIBShr transfer vectors
DISTRIBLIB/Include=(ACPTSV,CNCTND,CONFIG,CONFND,CONFTP,CVTUIR,CVTUQR,INITND)
DISTRIBLIB/Include=(INITSV,LINESV,NETKEY,NETNAM,NETOPN,NETPRS,NETWHO,NOPWD)
DISTRIBLIB/Include=(PRTNCB,RCVISV,REGNAM,REGNOD,SNDISV,STARSV,STRTND,TIMER)
DISTRIBLIB/Include=(TOPLND,TOPLSV)
!
GSMatch=LEQUAL,1,0                ! Use your own match criteria
Cluster=DISTRIBLIBShr_Transfer    ! Transfer vectors
Collect=DISTRIBLIBShr_Transfer,DISTRIBLIBShr_Transfer
!
Cluster=DISTRIBLIBShr_COMMONs     ! Fortran R/W COMMONs:
PSect_Attr=NETCOM,NOSHR           !    COMMON /NETCOM/
PSect_Attr=NETCON,NOSHR           !    COMMON /NETCON/
Collect=DISTRIBLIBShr_COMMONs,NETCOM
```

Figure 6. Modified Linker options file (`DistribLibShr.opt;2`).

transfer vectors source file as they do in the DistribLib User's Guide (not shown) for clarity. The resulting Macro-32 transfer vectors source file is shown in Figure 7.

The final step is to build the customized DistribLibShr shareable image and library.



```

        .Title  DISTRIBLIBShr_Transfer

        .Ident  /1.0/

; Transfer vectors file for DISTRIBLIBShr created at 27-JUN-1991 18:09:24.75

        .Macro  Transfer_Vector,Entry
        .Transfer Entry
        .Mask   Entry
        Jmp     L^Entry+2
        .EndM

        .PSect  DISTRIBLIBShr_Transfer,EXE,NOWRT,PIC,SHR,GBL

; Module CONFIGURE.for

        Transfer_Vector CONFIG  ; Transfer vector for CONFIG
        Transfer_Vector CONFTP  ; Transfer vector for CONFTP
        Transfer_Vector CONFND  ; Transfer vector for CONFND

; Module STARTUP.for

        Transfer_Vector INITSV  ; Transfer vector for INITSV
        Transfer_Vector STOPSV  ; Transfer vector for STOPSV
        Transfer_Vector INITND  ; Transfer vector for INITND
        Transfer_Vector STOPND  ; Transfer vector for STOPND

; Module SEND.for

        Transfer_Vector SNDISV  ; Transfer vector for SNDISV
        Transfer_Vector SNDRSV  ; Transfer vector for SNDRSV
        Transfer_Vector SNDCSV  ; Transfer vector for SNDCSV
        Transfer_Vector SNDUSV  ; Transfer vector for SNDUSV
        Transfer_Vector SNDIND  ; Transfer vector for SNDIND
        Transfer_Vector SNDRND  ; Transfer vector for SNDRND
        Transfer_Vector SNDCND  ; Transfer vector for SNDCND
        Transfer_Vector SNDUND  ; Transfer vector for SNDUND

; Module RECEIVE.for

        Transfer_Vector RCVISV  ; Transfer vector for RCVISV
        Transfer_Vector RCVRSV  ; Transfer vector for RCVRSV
        Transfer_Vector RCVCSV  ; Transfer vector for RCVCSV
        Transfer_Vector RCVUSV  ; Transfer vector for RCVUSV
        Transfer_Vector RCVIND  ; Transfer vector for RCVIND
        Transfer_Vector RCVRND  ; Transfer vector for RCVRND
        Transfer_Vector RCVCND  ; Transfer vector for RCVCND
        Transfer_Vector RCVUND  ; Transfer vector for RCVUND

; Module TIMER.for

        Transfer_Vector CLRTIM  ; Transfer vector for CLRTIM
        Transfer_Vector TIMER   ; Transfer vector for TIMER

        .End
```

Figure 7. Modified Macro-32 transfer vectors source file (DistribLibShr.vec;2).

Step 3: Re-build DistribLibShr

To build the customized DistribLibShr, type:

```
$ @DistribLibShr.bld
```

For more convenient use of the shareable image and its library, the system manager should modify the system-wide startup DCL command procedure to define logical names for the shareable image and its library, and install the shareable image, such as:

```
$ Assign /System DistribLibShr DistribLib
$ Assign /System DistribLib_Dir:DistribLibShr DistribLibShr
$ Assign /System "::Pub1:[Baker]" DistribLib_Dir:
$ Run Sys$System:Install
DistribLibShr /Open /Shared
```

By assigning `DistribLib` to `DistribLibShr`, user's can link their programs with `DistribLib/Library`, as with an object module library:

```
$ Link MyProgram,DistribLib/Library
```

The logical name `DistribLib_Dir:` is a convenience that allows a separate test version in another directory to be referenced by redefining a single logical name. (The "`::`" prefix restores the default DECnet node name, which allows the Linker to correctly locate `DistribLibShr.olb` even if it appears after a reference to a file that is on another node.) There is no need to install DistribLibShr writeable — it contains no read/write Fortran common blocks intended to be shared between programs.

5 MESSAGES

5.1 Informational Messages

The message below is the only message printed if MakeShare is invoked without any command line arguments.

%MAKESHARE-I-USAGE, Usage: @MakeShare library_name [shareable_image_name]

> *Explanation:* The command line usage summary is printed if MakeShare is invoked without any command line arguments.

The messages below are printed in the order given when MakeShare is invoked with a complete command line.

%MAKESHARE-I-START, Make of *shareable_image_name* started at *time*.

> *Explanation:* The command line arguments have been validated and MakeShare is ready to begin processing.

%MAKESHARE-I-LIBLIST, Creating Librarian listing file . . .

> *Explanation:* MakeShare is creating the Librarian listing file (`.lis`) of the object module library.

%MAKESHARE-I-OPTIONS, Creating Linker options file and Macro-32 transfer vectors source file . . .

Explanation: MakeShare is creating the Linker options file (`.opt`) for the default shareable image and also a Macro-32 transfer vectors source file (`.vec`) for the user to customize later.

%MAKESHARE-I-DCLPROC, Creating DCL command procedure file . . .

Explanation: MakeShare is creating the DCL command procedure file (`.bld`) to (optionally) assemble the transfer vectors, link the shareable image, create its library, and (optionally) install it.

%MAKESHARE-I-LINK, Creating default shareable image . . .

Explanation: MakeShare is creating the default shareable image file (`.exe`) and its map file (`.map`) without transfer vectors, with every entry point declared Universal, and every Fortran common block left shareable.

%MAKESHARE-I-OPTIONS, Appending Linker options for Fortran R/W COMMONs . . .

Explanation: MakeShare is adding commands to the end of the Linker options file (`.opt`) for the default shareable image in order to change the shareable (`SHR`) PSect attribute to non-shareable (`NOSHR`) for every Fortran common block.

%MAKESHARE-I-NOTES, Creating *shareable_image_name*.Release_Notes . . .

Explanation: MakeShare is creating the release notes file (`.Release_Notes`), which contains instructions for customizing the shareable image.

%MAKESHARE-I-COMPLETED, Make of *shareable_image_name* completed at *time*.

Explanation: MakeShare has either finished successfully or has been terminated by an error (either from DCL or a VMS utility).

5.2 Error Messages

The error messages below are generated when MakeShare finds an invalid command line argument. If an error is detected, MakeShare exits immediately after printing the message. Error messages may also be generated by DCL or by one of the VMS utilities used by MakeShare, such as the Librarian, the Linker, the Search utility, etc.

%MAKESHARE-F-NOSUCHFILE, Unable to find *library_name*.olb.

Explanation: MakeShare is unable to locate the object module library.

User Action: Correct the value of library_name and reenter the command.

%MAKESHARE-F-NAMETOOLONG, *shareable_image_name* is longer than 23 characters.

Explanation: The name part of shareable_image_name is too long.

User Action: Shorten the name supplied (or supply an acceptable name if the default was used), and reenter the command.

6 REFERENCES

[1] Digital Equipment Corp., 1990, *VMS DCL Dictionary: Part I*: Order no. AA-PBK5A-TE, Digital Equipment Corp., Maynard, Massachusetts.

[2] Digital Equipment Corp., 1990, *VMS DCL Dictionary: Part II*: Order no. AA-PBK6A-TE, Digital Equipment Corp., Maynard, Massachusetts.

[3] Digital Equipment Corp., 1988, *Guide to Using VMS Command Procedures*: Order no. AA-LA11A-TE, Digital Equipment Corp., Maynard, Massachusetts.

[4] Digital Equipment Corp., 1988, *VMS Install Utility Manual*: Order no. AA-LA29A-TE, Digital Equipment Corp., Maynard, Massachusetts.

[5] Digital Equipment Corp., 1988, *VMS Librarian Utility Manual*: Order no. AA-LA61A-TE, Digital Equipment Corp., Maynard, Massachusetts.

[6] Digital Equipment Corp., 1988, *VMS Linker Utility Manual*: Order no. AA-LA62A-TE, Digital Equipment Corp., Maynard, Massachusetts.

7 DISTRIBUTION

Machine-readable copies of MakeShare are available from the author:

Lawrence M. Baker
U.S. Geological Survey
345 Middlefield Road MS977
Menlo Park, California 94025
(415) 329-5608 or FTS 459-5608

APPENDIX A
SOURCE LISTING

```
$!==============================================================================
$!
$!
$! MakeShare.com -- Make a shareable image from an object module library.
$!
$! Usage:          @MakeShare library_name [ shareable_image_name ]
$!
$!                 where  library_name  is the name of the object library to be
$!                 bound, minus the trailing .olb, and shareable_image_name  is
$!                 the name to be given to the shareable image and its library,
$!                 also without any trailing file type designation (23  charac-
$!                 ters, maximum).  If shareable_image_name is not supplied, it
$!                 is constructed by adding an "Shr" suffix to library_name.
$!
$! Example:        @MakeShare PlotLib
$!
$!                 creates  shareable image PlotLibShr.exe from the modules and
$!                 entry points in PlotLib.olb.
$!
$!
$! Input file(s):  'P1'.olb    Input object library (P1 = library_name)
$!
$! Output file(s): 'P2'.lis    Librarian listing of 'P1'.olb contents
$!                                (P2 = shareable_image_name)
$!                 'P2'.bld    DCL command procedure to create 'P2'.exe, insert
$!                                'P2'.exe into a newly created image library,
$!                                'P2'.olb, and optional commands to install
$!                                'P2'.exe
$!                 'P2'.opt    Linker options file used with 'P2'.bld
$!                 'P2'.exe    Shareable image
$!                 'P2'.map    Linker map describing 'P2'.exe
$!                 'P2'.olb    Image library containing 'P2'.exe
$!                 'P2'.vec    Macro-32 source file containing transfer vectors
$!                                for all entry points in 'P2'.exe
$!                 'P2'.tmp    Temporary file from DCL Search command
$!                 'P2'.Release_Notes  Instructions for customizing 'P2'.exe
$!
$! Assumptions:    Librarian listings:
$!                    New modules are introduced with a line containing the
$!                       string "Module" in the first six characters.
$!                    There are no characters other than spaces surrounding the
$!                       module name after column seven.
$!                    Lines between "Module" lines are either all blank or
$!                       contain the names of the entry-points in the module.
$!                    There are no characters other than leading, trailing, and
$!                       separating spaces surrounding the entry-point names.
$!                    There are no other types of lines in the file once the
$!                       first "Module" line has been found.
$!                 Linker maps:
```

```
$!                    Fortran COMMONs are introduced with a line containing the
$!                       string "OVR,REL,GBL,  SHR,NOEXE,  RD,  WRT" in columns
$!                       92 through 125.
$!                    Columns 1 to 31 contain the corresponding program section
$!                       (PSect) name, followed by no characters other than
$!                       trailing spaces.
$!                    Multiple instances of the line may occur (on multiple
$!                       pages in the map file); if so, they are consecutive.
$!
$!                 These assumptions are true for VMS versions 3 through 5.
$!
$! Reference(s):    VAX-11 Linker Reference Manual, esp. Chapter 4  (VMS V3)
$!                 VAX/VMS Linker Reference Manual, esp. Section 3 (VMS V4)
$!                 VMS Linker Utility Manual, esp. Chapter 4       (VMS V5)
$!
$!
$! Author:          Lawrence M. Baker
$!                 U.S. Geological Survey
$!                 345 Middlefield Road  MS977
$!                 Menlo Park, CA  94025
$!
$!                                Disclaimer
$!
$! Although  this  program  has  been  used  by  the  U.S. Geological Survey, no
$! warranty, expressed or implied, is made by the USGS as to  the  accuracy  and
$! functioning  of  the program and related program material, nor shall the fact
$! of distribution constitute  any  such  warranty,  and  no  responsibility  is
$! assumed by the USGS in connection therewith.
$!
$!
$! Modification History:
$!
$! 26-Aug-1985  L. M. Baker     Remove UNIVERSAL=* option and replace with sep-
$!                                 arate Universal=entry_point options to be
$!                                 compatible with VMS V4.
$! 27-Jun-1991  L. M. Baker     Upgrade to VMS V5.
$!                              Require P1.
$!                              Add P2:  shareable_image_name.
$!                              Add Library /Create=(Blocks,Modules,Globals,
$!                                 History) options.
$!                              Add EMIT subroutines.
$!
$!
$!=============================================================================
$!
$!
$!             Subroutines EMIT_TOKEN, EMIT_SENTENCE, and EMIT_END
$!
$       Goto END_EMIT
$!
$ EMIT_TOKEN:           ! Subroutine to emit tokens to file "file" with right
$                       ! margin "margin," line prefix "prefix,", line suffix
$                       ! "suffix," and token separator "separator."  In addi-
$                       ! tion, the caller must Open/Write file <file_name> and
$                       ! set column = 0.
$                       ! To flush the last line before closing the file, call
$                       ! EMIT_END.
$!
```

```
$!
$! For example, to create lines of the form "LIBRARY/INCLUDE=(MODULE1,...)"
$!
$!      Open/Write file LINKER.OPTIONS
$!      margin = 80
$!      prefix = "LIBRARY/INCLUDE=("
$!      suffix = ")"
$!      separator = ","
$!      column = 0
$!      token = "MODULE1"
$!      GoSub EMIT_TOKEN
$!      :
$!      GoSub EMIT_END
$!      Close file
$!
$!
$       If (column .le. 0)
$       Then
$          newline = "T"
$          line = prefix + token
$          column = F$Length(prefix) + F$Length(token)
$       Else
$          If (column+F$Length(separator)+F$Length(token)+F$Length(suffix) -
              .gt. margin)
$          Then
$             line = line + suffix
$             Write file line
$             newline = "T"
$             line = prefix + token
$             column = F$Length(prefix) + F$Length(token)
$          Else
$             newline = "F"
$             line = line + separator + token
$             column = column + F$Length(separator) + F$Length(token)
$          EndIf
$       EndIf
$       Return
$!
$!
$ EMIT_END:             ! End of emit -- write the last line
$!
$       line = line + suffix
$       Write file line
$       Delete/Symbol column                    ! Debugging aid
$       Return
$!
$!
$ EMIT_SENTENCE:        ! Subroutine to emit sentences to file "file" by
$                       ! iteratively calling EMIT_TOKEN.
$!
$       j = 0
$       separator = "  "                        ! Separate the first word of a
$!                                              !   sentence with two spaces
$ EMIT_TOKEN_LOOP:
$!
$       token = F$Element (j," ",sentence)
$       j = j + 1
$       If (token .nes. " ")
$       Then
$          GoSub EMIT_TOKEN
$          separator = " "                      ! After the first word, use a
$          Goto EMIT_TOKEN_LOOP                 !    single space for separation
$       EndIf
$       Return
$!
$!
$ END_EMIT:
$!
$!==============================================================================
$!
$!                          Start of MAKESHARE
$!
$! Validate arguments
$!
$       If (P1 .eqs. "")
$       Then
$          Write Sys$Output ""
$          Write Sys$Output "%MAKESHARE-I-USAGE, ", -
              "Usage:  @MakeShare library_name [ shareable_image_name ]"
$          Write Sys$Output ""
$          Exit
$       EndIf
$!
$       If (F$Search("''P1'.olb") .eqs. "")
$       Then
$          Write Sys$Output ""
$          Write Sys$Output "%MAKESHARE-F-NOSUCHFILE, Unable to find ''P1'.olb."
$          Write Sys$Output ""
$          Exit
$       EndIf
$!
$       If (P2 .eqs. "")
$       Then
$          P2    = F$Parse (P1,,,"NAME","SYNTAX_ONLY") + "Shr"
$          P2SHR = P2
$       Else
$          P2SHR = F$Parse (P2,,,"NAME","SYNTAX_ONLY")
$       EndIf
$!
$       If (F$Length(P2SHR) .gt. 23)
$       Then
$          Write Sys$Output ""
$          Write Sys$Output -
              "%MAKESHARE-F-NAMETOOLONG, ''P2SHR' is longer than 23 characters."
$          Write Sys$Output ""
$          Exit
$       EndIf
$!
$!==============================================================================
$!
$       Write Sys$Output ""
$       Write Sys$Output -
           "%MAKESHARE-I-START, Make of ''P2' started at ''F$Time()'."
$       Write Sys$Output ""
$!
$       On Error Then $ Goto CLEANUP
```

```
$!
$!=============================================================================
$!
$! Obtain  a Librarian listing of the module names in the object module library.
$! Generate an /Include for each module found in the object  module  library,  a
$! transfer  vector entry for each global symbol in each module, a dummy GSMatch
$! option, and force all global symbols to be universal  symbols,  i.e.,  acces-
$! sible to the calling program when linking against the shareable image.
$!
$       Write Sys$Output -
           "%MAKESHARE-I-LIBLIST, Creating Librarian listing file . . ."
$!
$       Library /List='P2'.lis /Names /Width=80 'P1'.olb
$!
$       Write Sys$Output "%MAKESHARE-I-OPTIONS, Creating Linker options file", -
           " and Macro-32 transfer vectors source file . . ."
$!
$       Open/Read  file_1 'P2'.lis              ! Librarian listing
$       Open/Write file   'P2'.opt              ! Linker options file
$       Open/Write file_2 'P2'.vec              ! Transfer vectors (Macro-32)
$!
$!      Write file preambles
$!
$       Write file "!"
$       Write file "! Linker options file for ''P2SHR' created at ''F$Time()'"
$       Write file "!"
$       Write file "! Be sure to disable the Universal= options and enable", -
           " the GSMatch=,"
$       Write file -
           "! Cluster=, and Collect= options if you enable transfer vectors"
$       Write file "!"
$       Write file "!''P2'.obj,-                ! ''P2SHR' transfer vectors"
$!
$       Write file_2 ""
$       Write file_2 "  .Title  ''P2SHR'_Transfer"
$       Write file_2 ""
$       Write file_2 "  .Ident  /1.0/"
$       Write file_2 ""
$       Write file_2 -
           "; Transfer vectors file for ''P2SHR' created at ''F$Time()'"
$       Write file_2 ""
$       Write file_2 "  .Macro  Transfer_Vector,Entry"
$       Write file_2 "  .Transfer Entry"
$       Write file_2 "  .Mask   Entry"
$       Write file_2 "  Jmp     L^Entry+2"
$       Write file_2 "  .EndM
$       Write file_2 ""
$       Write file_2 "  .PSect  ''P2SHR'_Transfer,EXE,NOWRT,PIC,SHR,GBL"
$       Write file_2 ""
$!
$!=============================================================================
$!
$! Write  /Include=  options  from  scan of module names; write transfer vectors
$! from scan of entry points.  Save entry points for Universal= options.
$!
$       modules = 0
$       globals = 0
$       j = 0
$       margin = 80
$       prefix = "''P1'/Include=("
$       suffix = ")"
$       separator = ","
$!
$ SKIP_HEADER:          ! Skip to first module in file
$!
$       Read/End_of_file=EOF_1 file_1 junk
$       If (F$Extract(0,6,junk) .eqs. "Module") Then $ Goto NEXT_MODULE
$       Goto SKIP_HEADER
$!
$ LOOP_1:               ! Read the next line and determine if it is a new module
$!
$       Read/End_of_File=EOF_1 file_1 junk
$       If (F$Extract(0,6,junk) .eqs. "Module") Then $ Goto NEXT_MODULE
$       entry_points := 'junk'
$       If (entry_points .eqs. "") Then $ Goto LOOP_1
$       i = 0
$!
$ NEXT_ENTRY:           ! Create transfer vectors and a Universal= specification
$                       ! for all the entry points on a line
$!
$       entry = F$Element (i," ",entry_points)
$       i = i + 1
$       If (entry .eqs. " ") Then $ Goto LOOP_1
$       globals = globals + 1
$!      Remember entry points for construction of Universal= options below
$       entry_'globals' = entry
$       Write file_2 -
           "    Transfer_Vector ''entry'        ; Transfer vector for ''entry'"
$       Goto NEXT_ENTRY
$!
$ NEXT_MODULE:          ! Add module name to /Include= list, 8 at a time
$!
$       token := 'F$Extract(7,F$Length(junk)-7,junk)'
$       If (j .eq. 0) Then $ column = 0
$       GoSub EMIT_TOKEN
$       If (newline)
$       Then
$          j = 1
$       Else
$          j = j + 1
$          If (j .ge. 8)
$          Then
$             GoSub EMIT_END
$             j = 0
$          EndIf
$       EndIf
$       modules = modules + 1
$       Goto LOOP_1
$!
$ EOF_1:
$!
$       Close file_1
$!
$!      Write trailing /Include= list to Linker options file
$!
$       If (j .gt. 0) Then $ GoSub EMIT_END
```

```
$!
$!============================================================================
$!
$! Append Universal= options to Linker options file from entry_'i' symbols
$!
$       Write file "!"
$       j = 0
$       prefix = "Universal="
$       suffix = ""
$       separator = ","
$       i = 1
$!
$ LOOP_2:                 ! Add entry name to Universal= list, 8 at a time
$!
$       If (i .le. globals)
$       Then
$          token = entry_'i'
$          Delete/Symbol entry_'i'
$          i = i + 1
$          If (j .eq. 0) Then $ column = 0
$          GoSub EMIT_TOKEN
$          If (newline)
$          Then
$             j = 1
$          Else
$             j = j + 1
$             If (j .ge. 8)
$             Then
$                GoSub EMIT_END
$                j = 0
$             EndIf
$          EndIf
$          Goto LOOP_2
$       EndIf
$!
$!      Write trailing Universal= list to Linker options file
$!
$       If (j .gt. 0) Then $ GoSub EMIT_END
$!
$!============================================================================
$!
$! Append options to generate a shareable image to Linker options file
$!
$       Write file "!"
$       Write file "! Be sure to disable the Universal= options and enable", -
           " the GSMatch=,"
$       Write file -
           "! Cluster=, and Collect= options if you enable transfer vectors"
$       Write file "!"
$       Write file -
           "!GSMatch=LEQUAL,1,0         ! Use your own match criteria"
$       Write file "!Cluster=''P2SHR'_Transfer  ! Transfer vectors"
$       Write file "!Collect=''P2SHR'_Transfer,''P2SHR'_Transfer"
$       Write file "!"
$!
$       Close file
$!
$       Write file_2 ""
$       Write file_2 "  .End"
$!
$       Close file_2
$!
$!=============================================================================
$!
$! Create  a  DCL  command  procedure  to  perform the link and create the image
$! library, then use it to build an  image  without  any  modifications  to  the
$! existing PSect attributes.
$!
$       Write Sys$Output -
           "%MAKESHARE-I-DCLPROC, Creating DCL command procedure file . . ."
$!
$       Open/Write file 'P2'.bld                ! Commands to build image libr
$       Write file "$"
$       Write file "$ ! DCL command file for ''P2SHR' created at ''F$Time()'"
$       Write file "$"
$       Write file -
           "$ ! Be sure to edit ''P2'.opt if you enable transfer vectors"
$       Write file "$"
$       column = 0
$       prefix = ""
$       suffix = " -"
$       separator = " "
$       token = "$ !Macro"
$       GoSub EMIT_TOKEN
$       prefix = "!     "
$       token = "/List=''P2'.lis"
$       GoSub EMIT_TOKEN
$       token = "/Object=''P2'.obj"
$       GoSub EMIT_TOKEN
$       token = "''P2'.vec  ! Transfer vectors"
$       GoSub EMIT_TOKEN
$       suffix = ""
$       GoSub EMIT_END
$       column = 0
$       prefix = ""
$       suffix = " -"
$       token = "$ Link"
$       GoSub EMIT_TOKEN
$       prefix = "     "
$       token = "/Shareable=''P2'.exe"
$       GoSub EMIT_TOKEN
$       token = "/Map=''P2'.map"
$       GoSub EMIT_TOKEN
$       token = "''P2'.opt/Options"
$       GoSub EMIT_TOKEN
$       suffix = ""
$       GoSub EMIT_END
$!      Allow 50 bytes for each module/entry-point name to compute the size
$       blocks = ( ( ( modules + globals ) * 50 ) + 511) / 512
$       Write file "$ Library /Create=(Blocks:''blocks',Modules:''modules',", -
           "Globals:''globals',History:0) /Share -"
$       column = 0
$       suffix = " -"
$       token = "''P2'.olb"
$       GoSub EMIT_TOKEN
$       token = "''P2'.exe"
$       GoSub EMIT_TOKEN
$       suffix = ""
$       GoSub EMIT_END
$       Write file "$"
$       Write file "$ ! Enable the following commands if you need them:"
$       Write file "$"
$       column = 0
$       prefix = ""
$       suffix = " -"
$       token = "$ !Assign"
$       GoSub EMIT_TOKEN
$       prefix = "!     "
$       token = "''P2'.exe"
$       GoSub EMIT_TOKEN
$       token = "''P2SHR'"
$       GoSub EMIT_TOKEN
$       suffix = ""
$       GoSub EMIT_END
$       Write file "$ !Run Sys$System:Install"
$       column = 0
$       prefix = "!"
$       suffix = " -"
$       token = "''P2'.exe"
$       GoSub EMIT_TOKEN
$       prefix = "!     "
$       token = "/Open"
$       GoSub EMIT_TOKEN
$       token = "/Shared"
$       GoSub EMIT_TOKEN
$       token = "/Writeable"
$       GoSub EMIT_TOKEN
$       suffix = ""
$       GoSub EMIT_END
$       Write file "$"
$       Close file
$!
$!==============================================================================
$!
$       Write Sys$Output -
           "%MAKESHARE-I-LINK, Creating default shareable image . . ."
$!
$       @'P2'.bld                               ! Build default image & library
$!
$! The shareable image created above may be immediately installed if the
$! Fortran COMMONs are to be shared among all the referencing programs.  If,
$! however, a private copy of some or all of the Fortran COMMONs is required,
$! then the PSect attribute SHR must be changed to NOSHR using the PSect_Attr
$! Linker option in order for each COMMON to be forced into a user program.  To
$! aid in this process, we identify any PSects that look like Fortran COMMONs
$! and append a PSect_Attr=psect_name,NOSHR line for each one to the options
$! file 'P2'.opt created above.  The user may then modify this file to correctly
$! assemble the PSects into shareable and non-shareable image sections, and re-
$! create the shareable image and image library files.
$!
$       Write Sys$Output "%MAKESHARE-I-OPTIONS, Appending Linker options for", -
           " Fortran R/W COMMONs . . ."
$!
$!      Find Fortran COMMONs in map file
```

```
$!
$       Search /Exact /NoHeading /NoLog /NoNumbers /NoWindow /Format=PassAll -
           /Output='P2'.tmp 'P2'.map "OVR,REL,GBL,  SHR,NOEXE,  RD,  WRT"
$!
$       Open/Read   file_1 'P2'.tmp              ! Find Fortran COMMONs in map
$       Open/Append file 'P2'.opt                ! Add PSect_Attr=psect_name,
$       old_psect = ""                           !    NOSHR to Linker options
$       psect_count = 0                          !    file
$       Write file "Cluster=''P2SHR'_COMMONs     ! Fortran R/W COMMONs:"
$!
$ LOOP_3:
$!
$       Read/End_of_file=EOF_3 file_1 junk       ! Skip to next Fortran Common
$       If (F$Extract(91,34,junk) .nes. "OVR,REL,GBL,  SHR,NOEXE,  RD,  WRT") -
           Then $ Goto LOOP_3
$       new_psect = F$Extract(0,31,junk)
$!      To account for PSects that cross the map page boundaries
$       If (new_psect .eqs. old_psect) Then $ Goto LOOP_3
$       old_psect = new_psect
$       psect := 'new_psect'
$       Write file "PSect_Attr=''psect',NOSHR            !    COMMON /''psect'/"
$       psect_count = psect_count + 1
$       psect_'psect_count' = psect
$       Goto LOOP_3
$!
$ EOF_3:
$!
$       Close file_1
$       Delete 'P2'.tmp;0
$!
$       j = 0
$       prefix = "Collect=''P2SHR'_COMMONs,"
$       suffix = ""
$       separator = ","
$       i = 1
$!
$ LOOP_3A:               ! Add PSect name to Collect= list, 8 at a time
$!
$       If (i .lt. psect_count)
$       Then
$          token = psect_'i'
$          Delete/Symbol psect_'i'
$          i = i + 1
$          If (j .eq. 0) Then $ column = 0
$          GoSub EMIT_TOKEN
$          If (newline)
$          Then
$             j = 1
$          Else
$             j = j + 1
$             If (j .ge. 8)
$             Then
$                GoSub EMIT_END
$                j = 0
$             EndIf
$          EndIf
$          Goto LOOP_3A
$       EndIf
```

```
$!
$!      Write trailing Collect= list and close Linker options file
$!
$       If (j .gt. 0) Then $ GoSub EMIT_END
$!
$       Close file
$!
$!=============================================================================
$!
$       Write Sys$Output -
           "%MAKESHARE-I-NOTES, Creating ''P2'.Release_Notes . . ."
$!
$       Open/Write file 'P2'.Release_Notes
$       Write file ""
$       Write file "                          R E L E A S E   N O T E S"
$       Write file ""
$       column = 0
$       prefix = ""
$       suffix = ""
$       sentence = "''P2'.exe has been built and inserted into ''P2'.olb."
$       GoSub EMIT_SENTENCE
$       sentence = "Before it can be installed, all non-shared data areas," + -
           " e.g., Fortran read/write COMMONs, must be given the NOSHR" + -
           " attribute."
$       GoSub EMIT_SENTENCE
        sentence = "''P2'.opt has been created for you with all the Linker" + -
           " options necessary to build a new version of ''P2'.exe with all" + -
           " COMMONs marked NOSHR."
$       GoSub EMIT_SENTENCE
$       sentence = "You may edit ''P2'.opt to remove the PSect_Attr option" + -
           " for any COMMONs that are to be shared, i.e., that contain only" + -
           " read-only data, and to properly set the GSMATCH option."
$       GoSub EMIT_SENTENCE
$       sentence = "If you want to use transfer vectors in the new image," + -
           " edit ''P2'.bld and ''P2'.opt, and possibly the .Ident in" + -
           " ''P2'.vec (to match the GSMatch option)."
$       GoSub EMIT_SENTENCE
$       sentence = "To build a new version of ''P2'.exe, type"
$       GoSub EMIT_SENTENCE
$       GoSub EMIT_END
$       Write file ""
$       Write file "    $ @''P2'.bld"
$       Write file ""
$       column = 0
$       sentence = "You must define the logical name ''P2SHR' to refer to" + -
           " the image file if you do not move ''P2'.exe to Sys$Share:."
$       GoSub EMIT_SENTENCE
$       sentence = "Also, you must install the image to enable sharing of" + -
           " physical memory."
$       GoSub EMIT_SENTENCE
$       sentence = "(Use /Writeable only if shared read-write access is" + -
           " required.)"
$       GoSub EMIT_SENTENCE
$       sentence = "Sample commands are included (but commented out) in" + -
           " ''P2'.bld."
$       GoSub EMIT_SENTENCE
$       GoSub EMIT_END
$       Write file ""
$       sentence = "For additional information on creating and using" + -
            " shareable images, see the "
$       VMS_version = F$Extract (1,1,F$GetSYI("VERSION"))
$       If (VMS_version .le. 3) Then $ sentence = sentence + -
           "VAX-11 Linker Reference Manual, Chapter 4."
$       If (VMS_version .eq. 4) Then $ sentence = sentence + -
           "VAX/VMS Linker Reference Manual, Section 3."
$       If (VMS_version .ge. 5) Then $ sentence = sentence + -
           "VMS Linker Utility Manual, Chapter 4."
$       column = 0
$       GoSub EMIT_SENTENCE
$       GoSub EMIT_END
$       Write file ""
$       Close file
$!
$!============================================================================
$!
$       Write Sys$Output ""
$       Write Sys$Output "Read the release notes file ''P2'.Release_Notes to", -
           " customize and install ''P2SHR'."
$ CLEANUP:
$       Close/Error=CLOSE1 file
$ CLOSE1:
$       Close/Error=CLOSE2 file_1
$ CLOSE2:
$       Close/Error=DONE   file_2
$ DONE:
$       Write Sys$Output ""
$       Write Sys$Output -
           "%MAKESHARE-I-COMPLETED, Make of ''P2' completed at ''F$Time()'."
$       Write Sys$Output ""
$!
$!============================================================================
```