CERN Program Library Installation Guide

CERNLIB

Installation Guide

Unix, VMS and VM/CMS systems

Hints for: MVS, MSDOS and Windows/NT

Version 0.15 (November 1994)

Application Software and Databases Group

Computers and Network Division

CERN Geneva, Switzerland

```
CERN Program Library Office
CERN-CN Division
CH-1211 Geneva 23
Switzerland
Tel.  +41 22 767 4951
Fax.  +41 22 767 7155
Bitnet:  CERNLIB@CERNVM
DECnet:  VXCERN::CERNLIB (node 22.190)
Internet:  CERNLIB@CERN.CH
```


Contact Person:	CERN Program Library Office	(CERNLIB@CERN.CH)
Editor:	Jamie Shiers /CN	(Jamie.Shiers@CERN.CH)
LaTeX *expert*:	Michel Goossens /CN	(Michel.Goossens@CERN.CH)

Table of Contents

List of Tables

Part I

CERNLIB – Overview

Chapter 1: Intended audience

This manual is aimed primarily at those people responsible for the installation of the CERN Program Library on systems at CERN or elsewhere. Some of the material is only relevent for the CERN Program Library section of the CN/ASD group at CERN.

Having read this manual you should be able to:

1. Install all or part of the CERN Program Library on one of the supported platforms.
2. Understand how to port the Program Library to a new platform.

Certain sections of this manual will also be of interest to software developers, such as Appendix B, where the PATCHY flags are described and Appendix C, where reserved routine and common block prefixes are listed.

Chapter 2: Overview of the CERN Program Library and related environment

The CERN Program Library is a large collection of general purpose programs maintained and offered in both source and object code form on the CERN central computers. Most of these programs were developed at CERN and are therefore oriented towards the needs of a physics research laboratory. Nearly all, however, are of a general mathematical or data-handling nature, applicable to a wide range of problems.

2.1 Contents and Organization of the Library

The library contains several thousand subroutines and complete programs which are grouped together by logical affiliation into several hundred program packages. 80% of the programs are written in FORTRAN and the remainder in assembly code, or C usually with a FORTRAN version also available. The language supported is currently Fortran 77.

Each package is assigned a unique code, consisting of a letter and three or four digits. The letter is used to indicate the category in which the program or package resides. A package consists of one or more related subprograms with one package name and one or more user-callable entry names, all described briefly in a `Short write-up` [1], and if necessary, an additional `Long write-up`. The terms under which the library material and associated documentation may be distributed are given below.

2.2 Availability and Charging

- Access to the CERN Program Library is free of charge to all HEP users worldwide.
- Network access is the sole distribution method. In order to gain access, users must register the node name of their computer with the Program Library Office. This service is free of charge. Distribution on magnetic tape is not available.
- Non HEP academic and not-for-profit organizations are offered registered network access for one year for a registration fee of 1000 Swiss Francs. This service is free for physics departments of institutes in CERN member states.
- Commercial users are offered registered network access for 1 year for a base fee of 5000 Swiss Francs.

  The above fees are doubled for requests coming from non-member states.

The Library is not available for any purpose related to military applications.

Distribution to organizations not covered by the above rests at the discretion of the Director-General. Machine readable copies of the documentation are provided with the library and recipients are free to make copies for their users. Paper copies of the documentation are made available for a charge which covers the printing and handling cost. Charging is waived for cumulative orders of less than 100 Swiss Francs.

If your organisation is expected to pay a fee as described above, it will be billed by CERN after the material has been shipped. Please include special billing address and instructions, if any, with the request.

As of the publication date of this document the member states of CERN are Austria, Belgium, Czech Republic, Denmark, Finland, France, Germany, Greece, Hungary, Italy, The Netherlands, Norway, Poland, Portugal, Slovak Republic, Spain, Sweden, Switzerland and the United Kingdom.

States with Observer status are Israel, the Russian Federation, Turkey, Yugoslavia (status suspended after UN embargo, June 1992), the European Commission and Unesco.

2.3 Conditions of Use Outside CERN

Programs and documentation are provided solely for the use of the organization to which they are distributed, and may not be redistributed or reproduced in large numbers without the express agreement of CERN. Note that such agreement may have to be established somewhere else in addition to or instead of CERN in the case of programs originating from sources outside CERN. The appropriate Short Write-up gives information about authorship. The material cannot be sold. CERN should be given credit in all references, library documentation, and publications based on the programs.

If the programs are modified beyond what is necessary to adapt them to the local machine/system environment, it should be made clear in local documentation that they are locally modified versions of the CERN originals. CERN should be informed of such modifications, and given the possibility of introducing the same modifications in the original version. If local modifications are so important as to change significantly the behaviour of the program, its name should also be changed in order to avoid confusion with the original. CERN welcomes comments concerning the Program Library service, but undertakes no obligation for the maintenance of the programs, nor responsibility for their correctness, and accepts no liability whatsoever resulting from the use of its programs.

Chapter 3: CERNLIB installation environment

3.1 PATCHY

PATCHY is a source code management system that has been in use in High Energy Physics for many years. It is used for the maintenance, distribution and installation of almost all of the routines and packages that make up the CERN Program Library.

PATCHY and the associated auxiliary programs serve in development, maintenance, and inter-computer transport of source programs. Suitably structured source files containing several versions of a given program permit code selection and code modification (down to single-statement-level) by simple control cards to YPATCHY. Compacting and structuring of card files for efficiency (YTOBIN), maintenance of compacted files at the deck level (YEDIT), creation of machine-independent, transportable files (YTOCETA) and listing of compacted files (YLIST) and others are simple auxiliary operations in this environment.

Each of the PATCHY programs is a self-contained executable module. On Unix machines at CERN, these are typically found in the directory `/cern/pro/bin`.

3.2 FCASPLIT

FCASPLIT is a program that splits a file containing a mixture of routines in various languages, such as Fortran, C and Assembler (from whence the name is derived), into separate files. The file names are composed of the routine name with a suffix to identify the language, such as **.f** for Fortran etc. In addition, **FCASPLIT** will create both a shell script and a **MakeFile**, either of which may be used to compile the various routines. The default file extensions, listed below, are machine dependant.

The FCASPLIT header lines are automatically created by PATCHY on VMS and Unix machines for Fortran and C streams. The following PATCHY control statement shows how to request FCASPLIT header lines should they not be generated automatically.

Requesting FCASPLIT header lines

```
+ASM,34,R=! ./*DECK ID>, !.h */
```

Default file extensions (most Unix systems):

`.c`	C code
`.f`	Fortran code
`.s`	Assembler code

Default file extensions (VMS systems):

`.c`	C code
`.for`	Fortran code
`.mar`	Assembler code

Options:

`-f` Override the default name for the Fortran compiler
`-c` Override the default name for the C compiler
`-a` Override the default name for the Assembler
`-fo` Override the default options for the Fortran compiler
`-co` Override the default options for the C compiler
`-ao` Override the default options for the Assembler
`+fo` Add additional options to the defaults for the Fortran compiler
`+co` Add additional options to the defaults for the C compiler
`+ao` Add additional options to the defaults for the Assembler

Each routine must start with an identifying line :

Header lines for FCASPLIT

```
   CDECK  ID>,        in cols.  1-12  for Fortran
   /*DECK ID>,        in cols.  1-12  for C
   ;DECK  ID>,        in cols.  1-12  for assembler
    DECK  ID>,        in cols.  2-12  or
     DECK ID>,        in cols.  3-12  for anything else
            name   in cols. 13-40  gives the name
```

In the last two cases, or if *name* contains an extension, the file created will be *name* without extension .f, .c or .s added to it and without an entry into the script.

Example input file to FCASPLIT

```
CDECK  ID>, FMCLR.
      SUBROUTINE FMCLR
      INTEGER SYSTEMF
      IC = SYSTEMF('clear')
      END

/*DECK ID>, FMH.H*/
#define MAXNAME 8
#define MAXJOBN 8 /* Was 16 */
#define MAXHOST 8
#define MAXINFO 8 /* Was 16 */

/*DECK ID>, FAEXIT. */
void faexit_(icode)
int *icode;
{
   exit(*icode);
}
```

Running **FCASPLIT** on this input file produces the following output files:

Output from FCASPLIT

```
::::::::::::::
faexit.c
::::::::::::::
/*DECK ID>, FAEXIT. */
void faexit_(icode)
int *icode;
{
   exit(*icode);
}

::::::::::::::
fmclr.f
::::::::::::::
CDECK  ID>, FMCLR.
      SUBROUTINE FMCLR
      INTEGER SYSTEMF
      IC = SYSTEMF('clear')
      END

::::::::::::::
fmh.h
::::::::::::::
#define MAXNAME 8
#define MAXJOBN 8 /* Was 16 */
#define MAXHOST 8
#define MAXINFO 8 /* Was 16 */

::::::::::::::
temp.mkfca
::::::::::::::
ROUTINES = fmclr.o faexit.o

.f.o:
        xlf  -c -O -qextname -qcharlen=8192  $*.f

.c.o:
        cc  -c -O  $*.c

.s.o:
        as    $*.s

temp: $(ROUTINES)

::::::::::::::
temp.shfca
::::::::::::::
xlf  -c -O -qextname -qcharlen=8192  fmclr.f
cc  -c -O  faexit.c
```

Example of running FCASPLIT on an RS6000

```
zfatal:/home/cp/jamie/fatmen (42) fcasplit fmkuip.f
 FCASPLIT executing.
            Input file : fmkuip.f
          Shell script : fmkuip.shfca
             Make file : fmkuip.mkfca
        Fortran    name : xlf
        Fortran options : -c -O -qextname -qcharlen=8192
             CC    name : cc
             CC options : -c -O
      Assembler    name : as
      Assembler options :
  10315 lines written for    63 decks
     16 trailing comment lines ignored.
```

Example of running FCASPLIT on an HP

```
[csf] (358) /cern/pro/bin/fcasplit fmkuip.f
 FCASPLIT executing.
            Input file : fmkuip.f
          Shell script : fmkuip.shfca
             Make file : fmkuip.mkfca
        Fortran    name : f77
        Fortran options : -c -O -w +ppu
             CC    name : cc
             CC options : -c -O
      Assembler    name : as
      Assembler options :
  10315 lines written for    63 decks
     16 trailing comment lines ignored
```

Makefile generated by FCASPLIT on an HP

```
[csf] (359) more fmkuip.mkfca
ROUTINES = fatmen.o fmcd.o fmclr.o fmcopc.o fmcpc.o fmdumc.o  fmedit.o fmexit.o fmextr.o fmfc.o fmfndc.o fmgime.o fminic.o  fmkadd.o fmkadt.o fmkatt.o fmkcpl.o fmkdst.o fmkend.o fmkloc.o  fmkmod.o fmkmtp.o fmksrc.o fmkuwd.o fmld.o fmlogc.o fmlogn.o  fmlsc.o fmmakc.o fmmkdr.o fmmvc.o fmpwd.o fmquit.o fmrmc.o  fmrmdc.o fmrmtc.o fmrzld.o fmscat.o fmscpl.o fmsdst.o fmseac.o  fmsloc.o fmsmdk.o fmsmtp.o fmssrc.o fmstat.o fmsuwd.o fmtagc.o  fmtchc.o fmthrg.o fmtrek.o fmulcc.o fmupdc.o fmzoom.o zabend.o  fmslcc.o fmlokk.o fmsupd.o fmlnc.o fmrmlc.o fmnikk.o fmskan.o  fmklcc.o

.f.o:
        f77  -c -O -w +ppu  $*.f

.c.o:
        cc  -c -O  $*.c

.s.o:
        as    $*.s

fmkuip_all: $(ROUTINES)
```

Shell script generated by FCASPLIT on an HP

```
[csf] (360) more fmkuip.shfca
f77  -c -O -w +ppu  fatmen.f
f77  -c -O -w +ppu  fmcd.f
f77  -c -O -w +ppu  fmclr.f
f77  -c -O -w +ppu  fmcopc.f
f77  -c -O -w +ppu  fmcpc.f
f77  -c -O -w +ppu  fmdumc.f
f77  -c -O -w +ppu  fmedit.f
f77  -c -O -w +ppu  fmexit.f
f77  -c -O -w +ppu  fmextr.f
f77  -c -O -w +ppu  fmfc.f
f77  -c -O -w +ppu  fmfndc.f
f77  -c -O -w +ppu  fmgime.f
f77  -c -O -w +ppu  fminic.f
f77  -c -O -w +ppu  fmkadd.f
f77  -c -O -w +ppu  fmkadt.f
f77  -c -O -w +ppu  fmkatt.f
f77  -c -O -w +ppu  fmkcpl.f
f77  -c -O -w +ppu  fmkdst.f
f77  -c -O -w +ppu  fmkend.f
f77  -c -O -w +ppu  fmkloc.f
...
```

3.3 CMZ

CMZ is a source code management system that is compatible with **PATCHY** at the level of directives. Whereas **PATCHY** is oriented more to batch-like execution, **CMZ** provides also an interactive interface through which one may develop and install code. Most of the development performed by the CN/AS group is done using **CMZ**.

Chapter 4: Space requirements

The complete CERN library requires approximately 200MB of disk space. Slightly over 50MB may be saved if the sources are not kept locally.

Chapter 5: Description of CERNLIB components

The CERN Program Library consists of a number of independent Fortran callable libraries and a collection of complete programs. The overall structure is described briefly below. Note that the library is not static and small deviations from what is described may exist.

5.1 KERNLIB

KERNLIB is a Fortran callable library composed almost entirely of individual subroutines and functions. All of these are described in the CERNLIB short writeups manual [1].

The **KERNLIB**source files are organised as follows:

- Machine independent routines, mainly Fortran.
- Machine dependent routines, often C or Assembler.
- Numerical routines, including random number generators.

The actual contents of the library varies slightly from platform to platform, largely due to the machine dependent routines.

A complete list of routines, grouped according to section, is given below.

5.2 MATHLIB

MATHLIB is a Fortran callable library containing routines of a mathematical nature. It is made up of four components, namely

- BVSL - basic vector subroutine library. These routines were originally written for the IBM 3090 vector facility. However, a Fortran version is now available.
- GEN - general mathematical routines.
- LAPACK - the well-known linear algebra package.
- MPA - multiple precision floating point arithmetic routines.

Note that LAPACK and MPA may only be distributed in object form by CERN. Further details can be found in section P.4 and P.5.

5.3 PACKLIB

PACKLIB is a Fortran callable library made up of complete packages. A few of the packages, such as VMIO, are machine dependent. The bulk, however, run on a variety of systems including Unix, MS-DOS, Windows NT, VMS, VM/CMS and MVS.

Long writeups exist for each of these packages and these should be consulted for further details.

- EPIO - EP standard input/output package
- FFREAD - processing of Free Format data cards
- IOPACK - Input/output package for VM/CMS and MVS systems
- KAPACK - Key access package
- KUIP - Kit for a User interface package
- HBOOK - Histogramming, statistical analysis and Ntuple package
- MINUIT - Function minimization package
- VMIO - VM/CMS specific I/O package
- ZBOOK - Data structure management package
- ZEBRA - Data structure management package
- CDLIB - The API for the HEPDB detector geometry and calibration system
- FATLIB - The API for the FATMEN distributed file management system

5.4 The graphics libraries

The graphics libraries are divided into a kernel and driver specific libraries. Examples are shown below.

- GRAFLIB - the graphics kernel
- GRAFGKS - the GKS binding
- GRAFGDDM - the GDDM binding
- GRAFX11 - the X11 binding

Some of the above libraries may not be available on certain platforms.

5.5 PAWLIB

PAWLIB is the library used to build the various PAW modules. In addition to PAW itself, (apart from the main program), it also contains COMIS and SIGMA.

5.6 GEANT

GEANT is provided in a standalone library. The version number is contained in the library name.

5.7 The Monte Carlo libraries

The Monte Carlo libraries provide numerous event generators, such as the well-know Lund Monte Carlos, including those listed below. Again, the long writeup should be consulted for further details.

- COJETS - generator for high energy proton-proton and proton-antiproton collisions
- EURODEC - generator for high energy processes
- HERWIG - generator for hadron collisions
- ISAJET - the BNL generator for high energy proton-proton and proton-antiproton collisions
- JETSET - Lund Monte Carlo for jet fragmentation and e+e- anihilation
- PYTHIA - Lund Monte Carlo for high Pt hadron-hadron scattering
- LEPTO - Lund Monte Carlo for deep inelastic lepton-nucleon scattering
- PHOTOS - Monte Carlo for generation of radiative corrections in decays
- PDFLIB - Parton Density Functions
- TWISTER - Monte Carlo for QCD high Pt scattering
- FRITIOF - hadron-hadron, hadron-nucleus, nucleus-nucleus interactions
- ARIADNE - Lund Monte Carlo for QCD cascades in the Colour Dipole approximation

5.8 The CERNLIB modules

The CERNLIB modules consist of complete standalone programs, the most famous of which is surely PAW. A brief description of the various modules for a given system is given in the system dependent installation section of this manual.

Chapter 6: CERNLIB rules

The purpose of the rules described below is to improve the quality of the CERNLIB software whilst reducing the support effort.

6.1 Submission of material for installation

Material should preferably be submitted in one of the following formats:

- **PATCHY** *car* format.
- **CMZ** binary file.
- A flat file containing Fortran or C.
- A *tar* file containing a automatic build procedure, e.g. a makefile.

The CERN Program Library team will document a set of **PATCHY/CMZ** flags and *cpp* tags which should be used by developers.

If, for historical or other reasons, the standard flags cannot be used, an interface should be provided to ease the installation process.

The CERN Program Library team will provide a tool that will enable developers to request the installation of one or more packages in the **DEV** area.

The log files from such installations will be written into **/afs**.

A brief summary of the installation request will be returned by mail to the requestor.

The CERN Program Library team will provide a tool that will allow developers (and installers) to monitor the progress of the installation and see, in tabular form, what is installed in the **DEV**, **NEW** and **PRO** areas on the various platforms.

6.2 Movement of packages into the NEW area

Packages may be moved from the **DEV** to **NEW** area provided that a few basic criteria are met.

- The package(s) compile(s) on all *reference platforms*.
- The appropriate, automatic, tests are passed.
- A **README** file is provided describing the important changes.
- The CERN Program Library team reserves the right to add additional criteria in the light of experience.
- Packages will be moved on the reference platforms from **DEV** to **NEW** on a regular basis, typically overnight.
- The **NEW** area on the reference server, currently **asisftp.cern.ch**, will be updated from the reference platforms following checking of the installation logs and a posting of new features (e.g. the **README** file) to the **HEPLIB** distribution list.

6.3 Movement of packages from NEW to PRO

Packages will normally only move into the **PRO** area when a `Program Library Release` is performed.

Any new features relative to the previous release should be documented.

Certain packages, such as event generators, which are not maintained by the CN/ASD group, may move into **PRO** at the discretion of the CERN Program Librarian.

Routines that are in the **PRO** area should not generate any compilation warnings.

Testing of new releases should be performed both by the CERN Program Library team (the standard tests) and the developers (new features).

New or updated test procedures should be given to the CERN Program Library team as appropriate.

6.4 Bug fixes to PRO

Bug fixes to **PRO** will be made using **PATCHY** *cradles*.

The correction cradles must be relative to the current **PRO** version.

A test program which shows the effect of the bug (and fix) is to be provided.

A **README** file is to be provided.

Bug fixes will only be made if there is no reasonable workaround.

6.5 Release schedule

There shall normally be no less than two releases per calendar year and no more than four.

The approximate release date shall be announced (to HEPLIB and in the CERN news groups) at least **two months** in advance.

The approximate content shall be discussed and agreed at least **one month** in advance.

The exact content shall be discussed and agreed at least **two weeks** in advance.

The *final* versions of the various packages should be made available to the CERN Program Library team to permit installation in the public **NEW** area together with an appropriate announcement in the HEPLIB news group at least **one week** prior to the announced release date.

In the case of minor last minute corrections uncovered during user testing, the final code should be delivered at least **one working day** prior to the release.

In the event that such a deadline cannot be met, the CERN Program Librarian may decide to delay the release or to revert to a previous version of the package.

6.6 Documentation

Packages or routines must in all cases be documented.

The documentation should preferably be marked up in LaTeX, using the *cernman* style.

Should this not be the case, a postscript file should be provided.

Whereever possible, the documentation should be made available online, using the *World Wide Web*.

6.7 The reference environment

The reference environment shall be defined by the so-called *reference platforms* installed in the CERN Computer Centre.

These platforms will define things such as the *compiler release level*, *operating system level*, versions and revision levels of various products etc., that are used to generate the CERN Program Library.

The Program Library source and binaries will be made available via a reference server, currently **asis-ftp.cern.ch**.

Certain services in the CERN Computer Centre may decide to install private copies of the CERN Program Library (e.g. PARC, Shift, VXCERN).

These copies should in no way be considered to be the reference versions.

The CERN Program Library office will attempt to announce changes in the reference environment to HEPLIB at least 2 months in advance.

6.8 Support of various platforms

The supported platforms shall be defined by the *reference platforms*.

The CERN Program Library may decide to subcontract support for various operating systems or packages to outside institutes.

Platforms that are not in the reference environment nor subject to an agreement with an outside institute will be supported on a best-effort basis. Typically, this involves incorporating feedback from outside users who have installed the library on the appropriate platform (e.g. DESY for MVS).

6.9 Submission of ready-built packages

The CERN Program Librarian reserves the right to accept ready-built packages for redistribution. In such cases the author is responsible for providing a *tar* file containing ready-built libraries and/or executables for all packages that they are prepared to support.

Part II

CERNLIB – Initial Setup and Configuration

Chapter 7: The CERNLIB directory structure

The CERNLIB directory structure is basically the same for Unix, VMS and VM/CMS systems. We describe below the structure for Unix systems and note the differences for VMS and VM/CMS systems.

7.1 The CERNLIB tree

The directory tree for CERNLIB files begins at **/cern**. This may be a filesystem or a link. Below this top level directory are a number of subdirectories which fall into one of three categories:

- Packages, e.g. `CMZ, PATCHY` or `WWW`
- Directories relating to a specific release of the program library, e.g. `94a`
- Links to specific releases, as shown below.

The /cern directory tree

```
zfatal:/cern (76) ls /cern
93c     94a     cmz     new     patchy  pro
93d     WWW     mad     old     phigs
#
#
zfatal:/cern (80) ls -l
total 24
drwxrwxr-x   7 cpl-main asis         512 Jul 05 22:32 93c
drwxrwxr-x   6 cpl-main asis         512 Nov 10 18:13 93d
drwxr-xr-x   4 cernlib  asis         512 Oct 21 03:53 94a
drwxrwxr-x   3 cpl-gean asis         512 Feb 23 1993  WWW
drwxrwxr-x   5 cpl-hist asis         512 Oct 28 09:14 cmz
drwxrwxr-x   4 8120     asis         512 Nov 08 1992  mad
lrwxrwxrwx   1 cpl-main asis           3 Jun 14 13:16 new -> 93d
lrwxrwxrwx   1 cpl-main asis           3 Sep 23 09:02 old -> 93c
drwxrwxr-x   3 8117     asis         512 Nov 26 1992  patchy
drwxrwxr-x   4 cpl-main asis         512 Jun 10 15:24 phigs
lrwxrwxrwx   1 cernlib  asis           3 Oct 28 14:14 pro -> 93d
```

7.2 VMS specific details

On VMS systems, the logical name **CERN** points to the root directory for the CERNLIB tree, as shown below.

The CERN logical name

```
VXCRNA? sh log cern
   "CERN" = "_$22$DUS206:[CERN.]" (LNM$SYSTEM_TABLE)
```

This directory contains subdirectories as in the Unix case, as shown below.

CERNLIB subdirectory structure

```
VXCRNA? dir $22$dus206:[cern]
Directory $22$DUS206:[CERN]
000000.DIR;1       92B.DIR;1          93C.DIR;1          93D.DIR;1
94A.DIR;1          CMZ.DIR;1          DECW.DIR;1         FATMEN.DIR;1
GKS.DIR;1          HEPDB.DIR;1        HISTORIAN.DIR;1    HYDRA.DIR;1
LAPACK.DIR;1       MAD.DIR;1          NAG.DIR;1          NEW.DIR;1
OLD.DIR;1          PATCHY.DIR;1       PHIGS.DIR;1        PRO.DIR;1
RELEASE.LEVEL;8    WWW.DIR;1
```

7.2.1 VXCERN specific details

The CERN logical name

On VXCERN, the logical name **CERN** is a search list. It is composed of three individual logical names, as follows:

`CERNAXP`	The Alpha specific tree.
`CERNVAX`	The VAX specific tree.
`CERNNFS`	The **CERN** subdirectory of the asis system independant tree `/asis/share`, NFS mounted.

Defining the CERN logical names on VXCERN

```
$!
$! CERN library tree is on DISK$CERNLIB volume set
$!
$ if f$logical("disk$cernlib").nes.""
$ then
$   cernlib_disk = F$getdvi("disk$cernlib","FULLDEVNAM")
$   def/sys/exec/tran=(term,conc) cernaxp 'cernlib_disk'[cernaxp.]
$   def/sys/exec/tran=(term,conc) cernvax 'cernlib_disk'[cern.]
$ endif
$!
$! Now mount the /asis/share file system
$!
$ nfsmount/soft asisnfs:"/asis/share"                ASIS_SHARE
$ nfsdev=F$GETDVI("ASIS_SHARE","FULLDEVNAM")
$ If nfsdev.nes."" Then -
     assign/sys/exec/trans=(conc,term) 'nfsdev'[cern.] NFSCERN
$ If nfsdev.nes."" Then -
     assign/sys/exec/trans=(conc,term) 'nfsdev'[cern.] CERNNFS
$ Endif
$!
$! Now define the CERNlib logical names as a function of the above
$!
$ @cern:[new.mgr]cern_logicals
```

At the time of writing, some directories exist only in the **CERNVAX** tree. However, it is planned that all files shared between the **VAX** and **Alpha** architectures be moved progressively to the asis server.

N.B. the binaries and libraries are *incompatible* between VAX and Alpha systems. Thus, if you wish to copy the VAX executables over DECnet, use

COPY VXCERN::CERNVAX:[PRO.EXE]*.*

Similarly, to get the Alpha versions of the libraries, use

COPY VXCERN::CERNAXP:[PRO.LIB]*.*

The subdirectory structure

On VXCERN, the **NEW**, **PRO** and **OLD** subdirectories are *pointers* to other subdirectories, created using the VMS command **SET FILE/ENTER**.

Symbol definitions for CERNLIB products

On VXCERN, much of the system wide login procedure is performed using **CLUSTER$MANAGER:EXSYLOGIN.EXE**. This is done to speed up logins. The symbols definitions for the CERNLIB commands are defined in the program **CERN_ROOT:[MGR]CERNLOGIN.FOR**. You may either include this file into your own Fortran file or use the file **MAIN_CERNLOGIN.FOR**.

MAIN_CERNLOGIN.FOR

```
      IMPLICIT INTEGER(A-Z)
      INCLUDE '($LIBCLIDEF)'
      CHARACTER*(*) C_EXE
      PARAMETER(C_EXE='CERN_ROOT:[EXE]')
C
      I=LIB$K_CLI_GLOBAL_SYM
C
      INCLUDE 'CERN_ROOT:[MGR]CERNLOGIN.FOR'
C
      END
```

Symbol definitions on VXCERN using CERNLOGIN.FOR

```
C       IMPLICIT INTEGER(A-Z)
C       INCLUDE '($LIBCLIDEF)'
C       CHARACTER*(*) C_EXE
C       PARAMETER(C_EXE='CERN_ROOT:[EXE]')
C
C THIS IS A FAST VERSION OF THE USUAL SYSTEM WIDE LOGIN.COM .
C ALL CERN LIBRARY SYMBOL DEFINITIONS ARE CODED HERE.
C
C ==================================================================
C Customization section: enable if product is available at your site
C
      f_cmz  =1                ! CMZ      - CODEME
      f_garf =1                ! GARFIELD - CERN
      f_gks  =1                ! GKS      - GTS-GRAL
      f_his  =1                ! HISTORIAN - OPCODE
      f_mad  =1                ! MAD      - CERN
      f_nag  =1                ! NAG      - NAG Ltd
      f_patc =1                ! PATCHY   - CERN
      f_rzco =1                ! RZCONV   - CERN
      f_umon =1                ! UMON     - CERN
      f_vaxt =1                ! VAXTAP   - CERN
      f_www  =1                ! WWW      - CERN
C
C End of customization
```

```
C ====================================================================
C
C     I=LIB$K_CLI_GLOBAL_SYM
C
      RECODE=LIB$SET_SYMBOL('CERNLIB','$'//C_EXE//'CERNLIB' ,I)
      RECODE=LIB$SET_SYMBOL('CERN_LEVEL','PRO'                  ,I)
C
C---  PATCHY Symbols
C
      IF(F_PATC.EQ.1) THEN
        RECODE=
     +  LIB$SET_SYMBOL('FCASPLIT','$CERN:[patchy.4_15.EXE]FCASPLIT',I)
C
        RECODE=
     +  LIB$SET_SYMBOL('YCOM*PAR','$CERN:[patchy.4_15.EXE]YCOMPAR',I)
        RECODE=
     +  LIB$SET_SYMBOL('YEDI*T'  ,'$CERN:[patchy.4_15.EXE]YEDIT'  ,I)
        RECODE=
     +  LIB$SET_SYMBOL('YFRC*ETA','$CERN:[patchy.4_15.EXE]YFRCETA',I)
        RECODE=
     +  LIB$SET_SYMBOL('YIND*EX' ,'$CERN:[patchy.4_15.EXE]YINDEX' ,I)
        RECODE=
     +  LIB$SET_SYMBOL('YLIS*T'  ,'$CERN:[patchy.4_15.EXE]YLIST'  ,I)
C
        RECODE=
     +  LIB$SET_SYMBOL('YPAT*CHY','$CERN:[patchy.4_15.EXE]YPATCHY',I)
        RECODE=
     +  LIB$SET_SYMBOL('YSEA*RCH','$CERN:[patchy.4_15.EXE]YSEARCH',I)
        RECODE=
     +  LIB$SET_SYMBOL('YSHI*FT' ,'$CERN:[patchy.4_15.EXE]YSHIFT' ,I)
        RECODE=
     +  LIB$SET_SYMBOL('YTOBC*D' ,'$CERN:[patchy.4_15.EXE]YTOBCD' ,I)
        RECODE=
     +  LIB$SET_SYMBOL('YTOBI*N' ,'$CERN:[patchy.4_15.EXE]YTOBIN' ,I)
        RECODE=
     +  LIB$SET_SYMBOL('YTOC*ETA','$CERN:[patchy.4_15.EXE]YTOCETA',I)
C
        RECODE=LIB$SET_SYMBOL('YEXP*AND','@'//C_EXE//'YEXPAND',I)
C
      ENDIF
C
C---  CERNlib Tools
C
      RECODE=LIB$SET_SYMBOL('GETOPT','@'//C_EXE//'GETOPT'   ,I)
      RECODE=LIB$SET_SYMBOL('SETENV','@'//C_EXE//'SETENV " "',I)
      RECODE=LIB$SET_SYMBOL('XTERM' ,'@'//C_EXE//'XTERM " "',I)
      RECODE=LIB$SET_SYMBOL('RESIZE','@'//C_EXE//'RESIZE'   ,I)
      RECODE=LIB$SET_SYMBOL('MKCOMP*ILE','@'//C_EXE//'MKCOMPILE',I)
C
C---  CERNlib Products
C
      RECODE=LIB$SET_SYMBOL('DZEDIT','@'//C_EXE//'DZEDIT " "',I)
      RECODE=LIB$SET_SYMBOL('FM','$'//C_EXE//'FATMEN'       ,I)
      RECODE=LIB$SET_SYMBOL('HEPDB','$'//C_EXE//'HEPDB'     ,I)
      RECODE=LIB$SET_SYMBOL('GXINT','@'//C_EXE//'GXINT " "' ,I)
      RECODE=LIB$SET_SYMBOL('HIGZCONV','$'//C_EXE//'HIGZCONV',I)
      RECODE=LIB$SET_SYMBOL('KUIPC','$'//C_EXE//'KUIPC'     ,I)
```

```
      RECODE=LIB$SET_SYMBOL('PAW' ,'@'//C_EXE//'PAW.COM " "',I)
      RECODE=LIB$SET_SYMBOL('HTONEW','$'//C_EXE//'HTONEW'   ,I)
      RECODE=LIB$SET_SYMBOL('FATSREQ','$'//C_EXE//'FATSREQ' ,I)
      RECODE=LIB$SET_SYMBOL('SYSREQ','$'//C_EXE//'SYSREQ'   ,I)
      RECODE=
     +LIB$SET_SYMBOL('TELNETG','@'//C_EXE//'TELNETG.COM " "',I)
      RECODE=LIB$SET_SYMBOL('ZFTP','$'//C_EXE//'ZFTP'       ,I)
      RECODE=LIB$SET_SYMBOL('WYLBUR','$'//C_EXE//'WYLBUR'   ,I)
      RECODE=LIB$SET_SYMBOL('USE','$'//C_EXE//'WYLBUR'      ,I)
C
      RECODE=LIB$SET_SYMBOL('TO','$'//C_EXE//'NEWTONET.EXE' ,I)
C
C---  CMZ symbols
C
      IF(F_CMZ.EQ.1) THEN
      RECODE=LIB$SET_SYMBOL('CMZ' ,'@'//C_EXE//'CMZ.COM " "',I)
      ENDIF
C
C---  GARFIELD symbols
C
      IF(F_GARF.EQ.1) THEN
      RECODE=LIB$SET_SYMBOL('GA*RFIELD','$'//C_EXE//'GARFRUN',I)
C     RECODE=LIB$SET_SYMBOL('GH*ELP',
C    +                      'HELP/NOLIBLIST/LIBRARY=HELP$GARFIELD',I)
      RECODE=LIB$SET_SYMBOL('GED*IT','LSE/ENV=LSE$GARFIELD',I)
      ENDIF
C
C---  GKS symbols
C
C     IF(F_GKS.EQ.1) THEN
C
C     ENDIF
C
C---  HISTORIAN symbols
C
      IF(F_HIS.EQ.1) THEN
      RECODE=LIB$SET_SYMBOL('HISTE','$HISTORIAN_ROOT:[EXE]HISTE',I)
      RECODE=LIB$SET_SYMBOL('HISTG','$HISTORIAN_ROOT:[EXE]HISTG',I)
      RECODE=LIB$SET_SYMBOL('HISTO*R','$HISTORIAN_ROOT:[EXE]HISTOR',I)
      ENDIF
C
C---  MAD symbols
C
      IF(F_MAD.EQ.1) THEN
      RECODE=LIB$SET_SYMBOL('MAD8','@'//C_EXE//'MAD8.COM " "',I)
      ENDIF
C
C---  NAG symbols
C
      IF(F_NAG.EQ.1) THEN
C     RECODE=LIB$SET_SYMBOL('NAGH*ELP','$NAG_ROOT:[EXE]NAGHELP'    ,I)
      RECODE=LIB$SET_SYMBOL('NAGT*EST','@NAG_ROOT:[EXE]NAGTEST " "',I)
      ENDIF
C
C---  RZCONV symbols
C
      RECODE=LIB$SET_SYMBOL('RTOA','$'//C_EXE//'RTOA',I)
```

```
      RECODE=LIB$SET_SYMBOL('RTOX','$'//C_EXE//'RTOX',I)
      RECODE=LIB$SET_SYMBOL('RFRA','$'//C_EXE//'RFRA',I)
      RECODE=LIB$SET_SYMBOL('RFRX','$'//C_EXE//'RFRX',I)
C
C---  VAXTAP symbols
C
      IF(F_VAXT.EQ.1) THEN
       RECODE=LIB$SET_SYMBOL('EINIT'    ,'$'//C_EXE//'EINIT'    ,I)
       RECODE=LIB$SET_SYMBOL('LABELDUMP','$'//C_EXE//'LABELDUMP',I)
       RECODE=LIB$SET_SYMBOL('SETUP'    ,'$'//C_EXE//'SETUP'    ,I)
       RECODE=LIB$SET_SYMBOL('STAGE'    ,'$'//C_EXE//'STAGE'    ,I)
       RECODE=LIB$SET_SYMBOL('TAPECOPY' ,'@'//C_EXE//'TAPECOPY' ,I)
       RECODE=LIB$SET_SYMBOL('WRTAPE'   ,'$'//C_EXE//'WRTAPE'   ,I)
       RECODE=LIB$SET_SYMBOL('XTAPE'    ,'$'//C_EXE//'XTAPE'    ,I)
      ENDIF
C
C---  WWW symbols
C
      IF(F_WWW.EQ.1) THEN
      RECODE=LIB$SET_SYMBOL('WWW'     ,'$CERN:[WWW.EXE]WWW'        ,I)
      RECODE=LIB$SET_SYMBOL('WEB'     ,'@CERN_ROOT:[EXE]WEB " "'   ,I)
C     RECODE=LIB$SET_SYMBOL('XMOSAIC' ,'@CERN:[WWW.EXE]XMOSAIC'    ,I)
      ENDIF
C
```

7.3 VM/CMS specific details

7.3.1 CERNVM specific details

Chapter 8: Initial setup and configuration for Unix systems

The following steps must be followed before installing CERNLIB software on Unix systems.

1. Create a directory for the CERN software
2. Install the PATCHY modules. For further information, see chapter 15.1
3. Install the installation procedures. For further information, see chapter 16
4. Define the appropriate environmental variables. For further information, see chapter 16.

Chapter 9: Initial setup and configuration for VMS systems

The following steps must be followed before installing CERNLIB software on VMS systems.

1. Create a directory for the CERN software
2. Install the PATCHY modules
3. Install the installation command files
4. Define the appropriate symbols

9.1 Installing PATCHY

The PATCHY modules are stored in the **BACKUP** saveset **VXCERN::CERN:[PRO.BCK]CRNPAT.BCK**, or on asis in the backup directory for a given release, e.g. **cernlib/vax_vms/94a/bck.** We can either retrieve and unpack this saveset or install PATCHY from scratch as shown below.

Installing PATCHY from scratch

```
VSCLIB? create/directory [.patchy]

VSCLIB? create/directory [.scratch] ! work directory for patchy installation

VSCLIB? set default [.patchy]

VSCLIB? copy vxcern::cernvax:[patchy.src.vaxvms]*.* *

VSCLIB? edit/edt patchy.com  ! modify the source, work and target directories as appropriate

VSCLIB? @patchy
```

N.B. Ensure that you modify all lines with <<< MODIFY THIS >>> on them in the example below.

PATCHY.COM command file

```
$!******************************************************************
$!*                                                                *
$!*  Patchy installation job                                       *
$!*                                                                *
$!******************************************************************
$!
$!--- Customization section ----------------------------------------
$! SDIR="CERN:[PATCHY.SRC.VAXVMS]"     ! Patchy source directory
$! WDIR="DISK$SCRATCH:[PUBCR.WORK]"    ! working directory
$  sdir="disk$user1:[jamie.patchy]"    ! <<< MODIFY THIS >>>
$  wdir="disk$user1:[jamie.scratch]"   ! <<< MODIFY THIS >>>
$!------------------------------------------------------------------
$!
$ ver_imag=F$ENVIRONMENT("verify_image")
$ ver_def =F$ENVIRONMENT("DEFAULT")
```

```
$ ver_proc=F$VERIFY(0)
$
$ SET DEFAULT 'WDIR'
$
$ If F$SEARCH("P4INCETA.CET").eqs."" Then COPY 'SDIR'P4INCETA.CET *.*
$!-------------
$ If p1.nes."R"
$ Then
$   @'SDIR'rceta.sh
$   differ p4boot.com 'SDIR'p4boot.sh0
$   @p4boot 0
$ Else
$   copy 'SDIR'p4boot.sh0 p4boot.com
$   @p4boot 1
$   @p4boot 2
$ Endif
$!-----------------------------------------------------------------------
$ copy/log   Y*.EXE   CERN_ROOT:[EXE] ! <<< MODIFY THIS >>>
$ delete/log Y*.EXE;*,RCETA.*;*
$!
$ dummy=F$VERIFY(ver_proc,ver_imag)
$ SET DEFAULT 'ver_def'
$ Exit
```

9.2 Creating directories for CERNLIB

The following example shows how one might setup the CERNLIB tree on a new system.

Example of directory setup for installing the CERNLIB software

```
VSCLIB? show default

  DISK$USER1:[CERNLIB]

VSCLIB? create/directory [.cern]

VSCLIB? show logical disk$user1

   "DISK$USER1" = "_UXCNB1$DUA20:" (LNM$SYSTEM_TABLE)

VSCLIB? def/job/tran=(conc,term) cern _uxcnb1$dua20:[cernlib.cern.]

VSCLIB? create/directory cern:[cmz]              ! If you wish to install CMZ

VSCLIB? set default cern:[new]

VSCLIB? create/directory cern:[new.src]

VSCLIB? create/directory cern:[new.src.car]      ! For the source files

VSCLIB? create/directory cern:[new.mgr]          ! for the installation command files

VSCLIB? create/directory cern:[new.exe]          ! required for GETOPT, SETENV etc.
```

```
VSCLIB? create/directory cern:[new.lib]          ! for OLBs and link options files
```

9.3 Retrieving the installation command files

We now retrieve the installation files as shown below.

Retrieving the installation files

```
VSCLIB? set default cern:[new.mgr]

VSCLIB? copy vxcern::cern:[new.mgr]*.com; *

VSCLIB? copy vxcern::cern:[new.mgr]*.for; *

VSCLIB? copy vxcern::cern:[new.mgr]*.c; *

VSCLIB? set default [-.exe]

VSCLIB? copy vxcern::cern:[new.exe]*.com; *

VSCLIB? set default cern:[new.lib]

VSCLIB? copy vxcern::cern:[new.lib]*.opt; *

VSCLIB? create/directory cern:[decw] ! If you want to link PAW with the old version
                                     ! of DECwindows
VSCLIB? set default cern:[decw]

VSCLIB? copy vxcern::cernvax:[decw]*.exe *
```

We now compile and link the program **MAIN_CERNLOGIN.FOR** and customise our **LOGIN.COM**.

Creating MAIN_CERNLOGIN.EXE

```
VSCLIB? fortran cern:[new.mgr]main_cernlogin

VSCLIB? link main_cernlogin
```

Before running this program, we must define the logical name **CERN_ROOT**. Note that CERN_ROOT should be defined consistantly with the CERN logical name. In a production environment, CERN_ROOT is typically the PRO subdirectory of the CERN tree. When installing, however, it is wise to point CERN_ROOT to the NEW area, as shown below.

Defining CERN_ROOT

```
VSCLIB? define/sys/exec/trans=(concealed,terminal) -
cern_root -
disk$user1:[jamie.cernlib.cern.new.]
```

Example LOGIN.COM for the CERNLIB account

```
$ define cern _uxcnb1$dua20:[cernlib.cern.] /trans=(conc,term)
$ run cern:[new.mgr]main_cernlogin
$!
$! The following symbols are defined by
$! MAIN_CERNLOGIN but we chose to keep
$! these modules in a different place
$!
$ ycompare :== $cern:[new.exe]ycompare
$ yedit    :== $cern:[new.exe]yedit
$ yfrceta  :== $cern:[new.exe]yfrceta
$ yindex   :== $cern:[new.exe]yindex
$ ylist    :== $cern:[new.exe]ylist
$ ypatchy  :== $cern:[new.exe]ypatchy
$ ysearch  :== $cern:[new.exe]ysearch
$ yshift   :== $cern:[new.exe]yshift
$ ytobcd   :== $cern:[new.exe]ytobcd
$ ytobin   :== $cern:[new.exe]ytobin
$ ytoceta  :== $cern:[new.exe]ytoceta
$!
$ yexp*and :== @cern:[new.exe]yexpand
$!
$ fcasplit :== $cern:[new.exe]fcasplit
$!
$ cern_level :== new
$!
$ @cern:[new.mgr]plienv
```

9.4 Retrieving the source files

We can retrieve the source files as follows.

Retrieving the source files

```
VSCLIB? set default cern:[new.src.car]

VSCLIB? copy vxcern::cern:[new.src.car]*.c%% *  ! .cra and .car files
```

If we are in DECnet areas 22 or 23, the DECnet areas reserved for CERN, the installation procedures will automatically access the sources through VXCERN.

9.5 Final configuration

We are now ready to begin the CERNLIB installation. If there has been no previous installation of CERNLIB on this system, we must procede as follows:

1. Compile **GETHOSTNAME.C** and place the object file in **CERN:[NEW.LIB]**.

2. Prior to building **PACKLIB**, we must build

`CERNLIB`	The **CERNLIB** command. Built by typing **make cernlib**.
`FCASPLIT`	A PATCHY auxiliary. Built by typing **make fcasplit**.
`KUIPC`	The KUIP compiler. Built by typing **make kuipc**.

For further information, see chapter **??**.

Chapter 10: Initial setup and configuration for VMS systems when *asisnfs* is accessible

For systems at CERN, it may be more appropriate to access the source files over **NFS** from asisnfs. In this case, we procede as above *except* that we do not need to copy the **.cra** and **.car** files from VXCERN or asisftp. Instead, we modify our **LOGIN.COM** as shown below to NFS mount the appropriate asis file system.

CERNLIB LOGIN.COM to access sources over NFS from asisnfs

```
$!
$! Command file to define environment for CERNLIB installation
$! on AXCLIB (Alpha system)
$!
$! N.B. before running this command file, the following must have
$! been performed:
$!
$! 1) The root for the CERNLIB installation must be setup.
$!
$!    On the CNLAVC, the CERNLIB tree is installed on
$!    VSCLIB$USER1.
$!
$!    e.g. CREATE/DIRECTORY VSCLIB$USER1:[CERNLIB]
$!
$!         CREATE/DIRECTORY VSCLIB$USER1:[CERNLIB.CERNAXP]
$!
$!    These directory names are arbitrary. One could equally
$!    well use
$!
$!         CREATE/DIRECTORY VSCLIB$USER1:[FRODO]
$!
$!         CREATE/DIRECTORY VSCLIB$USER1:[FRODO.BAGGINS]
$!
$! 2) The subdirectory for the appropriate version must be created.
$!
$!    e.g. CREATE/DIRECTORY VSCLIB$USER1:[CERNLIB.CERNAXP.94A]
$!
$!    Again, this directory name is arbitrary. One could have used
$!    NEW, 93D, GLONK etc.
$!
$! 3) The subdirectories [.MGR] and [.EXE] must be created.
$!
$!    e.g. CREATE/DIRECTORY VSCLIB$USER1:[CERNLIB.CERNAXP.94A.MGR]
$!
$!    e.g. CREATE/DIRECTORY VSCLIB$USER1:[CERNLIB.CERNAXP.94A.EXE]
$!
$! 4) The command files in VXCERN::CERN_ROOT:[EXE] and
$!                         VXCERN::CERN_ROOT:[MGR] must be copied
$!    to the appropriate subdirectory.
$!
$! 5) Finally, the files VXCERN::CERN_ROOt:[MGR]*.FOR should
$!    be retrieved and compiled. MAIN_CERNLOGIN.FOR includes
$!    CERN_ROOT:[MGR]CERNLOGIN.FOR. As CERN_ROOT is not defined
$!    at this stage, simply edit the file and include CERNLOGIN.FOR
$!    directly.
$!
```

```
$!  If you cannot access the CERN sources via NFS, the following
$!  steps are also required.
$!
$!  a) Create the subdirectories [.SRC] and [.SRC.CAR]
$!
$!  b) Copy the .CAR and .CRA files from asisftp or VXCERN
$!     to the [.SRC.CAR] directory.
$!
$!  c) Remove all references to NFS below.
$!
$! ================================================================
$!
$! Mount /asis/share tree using NFS. This avoids us having to copy
$! the .CAR and .CRA files locally
$!
$ if f$trnlnm("asis_share") .eqs. "" then nfsmount/soft asisnfs:"/asis/share" -
     asis_share
$ nfsdev     = f$getdvi("ASIS_SHARE","FULLDEVNAM")
$ define/sys/exec nfscern 'nfsdev'[cern.] /trans=(conc,term)
$!
$! Set the CERNLIB level
$!
$ cern_level :== 94a
$!
$! Define the CERN and CERN_ROOT logical names
$!
$ define/SYS/EXEC/TRANS=(CONCEALED,TERMINAL) -
     cern _$177$dka300:[cernlib.cernaxp.],'nfsdev'[cern.]
$!
$ define/SYS/EXEC/TRANS=(CONCEALED,TERMINAL) -
     cern_root _$177$dka300:[cernlib.cernaxp.'cern_level'.],-
     'nfsdev'[cern.'cern_level'.]
$!
$! For Alpha, need also CERNAXP
$!
$ define/SYS/EXEC/TRANS=(CONCEALED,TERMINAL) -
     cernaxp _$177$dka300:[cernlib.cernaxp.]
$!
$! Define various symbols required by the CERNLIB installation
$! (and CERNLIB users)
$!
$ run main_cernlogin
$!
$! Reassert the CERNLIB level (MAIN_CERNLOGIN sets it to PRO)
$!
$ cern_level :== 94a
$!
$! Set the Program Library Installation Environment
$!
$ @cern:['cern_level'.mgr]plienv
$!
$! Override the settings of the following symbols made
$! by MAIN_CERNLOGIN.EXE
$!
$! We must also install these files...
$!
$ ycompare   :== $cern:['cern_level'.exe]ycompare
```

```
$ yedit      :== $cern:['cern_level'.exe]yedit
$ yfrceta    :== $cern:['cern_level'.exe]yfrceta
$ yindex     :== $cern:['cern_level'.exe]yindex
$ ylist      :== $cern:['cern_level'.exe]ylist
$ ypatchy    :== $cern:['cern_level'.exe]ypatchy
$ ysearch    :== $cern:['cern_level'.exe]ysearch
$ yshift     :== $cern:['cern_level'.exe]yshift
$ ytobcd     :== $cern:['cern_level'.exe]ytobcd
$ ytobin     :== $cern:['cern_level'.exe]ytobin
$ ytoceta    :== $cern:['cern_level'.exe]ytoceta
$!
$ yexp*and   :== @cern:['cern_level'.exe]yexpand
$!
$ fcasplit   :== $cern:['cern_level'.exe]fcasplit
$!
$! Private symbols
$!
$ ed*it      :== edit/tpu/section=sys$login:eve
$ t*ype      :== type/page
```

Part III

CERNLIB – Software Installation Guide

Chapter 12: Installing ready-built libraries and modules

This chapter describes how to retrieve and install ready-built libraries and modules from the asis server at CERN. More details on asis can be found in section H.

12.1 Retrieving tar files for Unix systems

Compressed *tar* files are kept in the directory cernlib/@sys/pro/tar, where @**sys** should be replaced by the Transarc name for your system, e.g. **rs_aix32** for the RS6000 running AIX 3.2. Examples of the Transarc naming convention are given in table E on page 113. This directory contains a number of files with the extension **.contents**, each of which describes the contents of the corresponding file with extension **.tar.gz**.

N.B. the *tar* files are only created when a release is made (typically a few days after the release). If you wish to install a version of the CERN library that has *not yet been released* please follow the instructions given in 13.

`cernbin.contents`	The binaries from the **bin** directory, e.g. **pawX11**.
`cernglib.contents`	The graphics libraries, e.g. libgrafGKS.a, libgrafX11.a, libgraflib.a, libpawlib.a
`cernlib.contents`	Libraries such as libkernlib.a, libpacklib.a, libmathlib.a and libphtools.a.
`cernmgr.contents`	The CERN **mgr** tree, required if you wish to reinstall all or part of CERNLIB locally.
`cernsrc.contents`	The contents of the **src** directory, i.e. the **.car** and **.cra** files.
`cmz.contents`	The **CMZ** distribution kit.
`gcalor.contents`	Various cross section files used by GEANT.
`geant315.contents`	The GEANT 3.15 distribution.
`mclibs.contents`	The Monte-Carlo libraries, e.g. JETSET, PHOTOS etc.
`patchy.contents`	The PATCHY distribution kit.

12.1.1 Retrieving the complete distribution

Procede as follows:

1. Connect to asisftp.cern.ch via **ftp**. Use username **anonymous**, password **your e-mail address**, e.g. jamie@zfatal.cern.ch.

2. Go to the appropriate directory for your machine type and the release that you wish to retrieve, e.g. **cd cernlib/hp700_ux90/94a/tar**. You will see a message like the following:

Welcome message

```
ftp> cd 93d/tar
250-
250- This directory contains compressed files of CERNlib release 93d for HP/UX 9.0
250-
250- Files ending in .tar.gz have been compressed using gzip. gzip/gunzip for
250- HP/UX are available in this directory in the gzip.tar file. Get this first
250- and untar in a directory in the search path. Also take a new copy of
250- plitar; this will use gzip -d ( equivalent to gunzip ) to uncompress files
```

```
250- ending in .tar.gz.
250-
250-
250-Please read the file README
250-  it was last modified on Thu Nov  4 18:16:40 1993 - 83 days ago
250 CWD command successful.
ftp>
```

3. Follow the instructions given in the welcome message, e.g. **retrieve and read any README files**.
4. Go to **binary** mode and retrieve all compressed tar files.
5. Unpack using **plitar**.

12.1.2 Retrieving the source files

The complete sources can be obtained by retrieving and unpacking only the cernsrc.tar.gz file. Assuming that you wish to retrieve individual sources, procede as follows:

1. Connect to asisftp.cern.ch using anonymous *ftp* as shown above.
2. Change directory to **cernlib/share/94a/src/car** (or the corresponding directory for the appropriate release).
3. Retrieve the **.car** files of interest, e.g. **paw.car**, **hbook.car**, etc.

12.1.3 Retrieving the libraries

One may obtain all of the libraries or individual sets as shown above. To obtain individual libraries only, procede as follows:

1. Connect to asisftp.cern.ch using anonymous *ftp* as shown above.
2. Change directory to **cernlib/@sys/94a/lib** (or the corresponding directory for the appropriate release).
3. Switch to **binary** mode.
4. Retrieve the **.a** files of interest, e.g. **libpacklib.a**, **libpawlib.a**.

12.1.4 Retrieving the binaries

One may obtain all of the binaries as shown above. shown above. To obtain individual binaries only, procede as follows:

1. Connect to asisftp.cern.ch using anonymous *ftp* as shown above.
2. Change directory to **cernlib/@sys/94a/bin** (or the corresponding directory for the appropriate release).
3. Switch to **binary** mode.
4. Retrieve the files of interest, e.g. **pawX11**, **kxterm**, etc.

12.2 Retrieving backup savesets for VMS systems

The backup savesets for VMS systems are currently compressed using the `gzip` program. If you have access to this program, it is more efficient in terms of network load and CPU load on the `asisftp.cern.ch` server to transfer the compressed file and uncompress it on your machine.

If you do **not** have access to this program, simply leave of the `.gz` suffix when performed the transfer and the files will be uncompressed on the fly.

12.2.1 GZIP/GUNZIP

The executables for GZIP and GUNZIP may be found in VXCERN::USR:[LOCAL.EXE] (VAX/VMS) and VXCERN::USR:[LOCALAXP.EXE] (OpenVMS).

Unlike Unix systems, which add an extension .gz when compressing a file, -GZ is appended to the file name.

Running GZIP or GUNZIP on VMS systems

```
AXCRNC? gzip -v login.com

DISK$CD:[JAMIE]LOGIN.COM;1: 61.8% -- replaced with DISK$CD:[JAMIE]LOGIN.COM-gz

AXCRNC? gunzip -v login.com

DISK$CD:[JAMIE]LOGIN.COM-gz: 61.8% -- replaced with DISK$CD:[JAMIE]LOGIN.COM
```

The following savesets were built for release 94A:

`crncmz.bck.gz`	CMZ
`crnlib.bck.gz`	The .OLBs (KERNLIB, MATHLIB, PACKLIB etc.) and .EXEs.
`crnpat.bck.gz`	The PATCHY executables.
`crnsrc.bck.gz`	The source files.

Post 94A releases will be made in the same way as for Unix systems, i.e.

`cernbin.tar.gz`	The modules (binaries)
`cernglib.tar.gz`	The graphics libraries
`cernlib.tar.gz`	The non-graphics libraries
`cernmgr.tar.gz`	The installation scripts
`cernsrc.tar.gz`	The source files (.CAR and .CRA)
`cmz.tar.gz`	CMZ
`geant321-src.tar.gz`	GEANT 3.21 source
`geant321.tar.gz`	GEANT libraries etc.
`mclibs.tar.gz`	The Monte Carlo libraries
`patchy.tar.gz`	The PATCHY modules

N.B. Having retrieved (and uncompressed) the savesets, the correct block size must be set using the RESIZE command, available from VXCERN::CERN:[PRO.EXE]RESIZE.COM.

Setting the correct blocksize

```
AXCRNC? resize -s 32256 crnlib.bck
resize: setting record size of CRNLIB.BCK to 32256 bytes...
AXCRNC?
```

On systems running VMS 6.1 or higher, the resize command should be replaced by the **SET FILE/ATTRIBUTES** command, e.g.
SET FILE/ATTRIBUTES=LRL=32256 CRNLIB.BCK
With version 3.3 (February, 1994) of TGV Multinet TCP/IP software for OpenVMS, it is possible to set the record size for binary transfers in FTP (previously the only allowed sizes were 512 and 2048 bytes). The commands are RECORD-SIZE nnnnn when the ftp session originates on the openVMS computer, and QUOTE SITE RMS RECSIZE nnnnn when ftp is started from the non-VMS end. Here are examples of the two cases.

For a transfer starting on an OpenVMS computer

```
$ FTP SHIFT.CERN.CH
SHIFT.CERN.CH> USER yourname
<Password required for yourname.
Password:
<User leeiv logged in.
SHIFT.CERN.CH> VERSION
DUKPHY.PHY.DUKE.EDU MultiNet FTP user process 3.3(109)
SHIFT.CERN.CH> BINARY
Type: Image, Structure: File, Mode: Stream
SHIFT.CERN.CH> RECORD-SIZE 32400
SHIFT.CERN.CH> RECORD-SIZE
Record size for IMAGE files: 32400
SHIFT.CERN.CH> GET myfile.rzhist myfile.rzhist
SHIFT.CERN.CH> QUIT
$
```

For a transfer starting from a non-Multinet computer

```
> ftp dukphy.phy.duke.edu
Connected to dukphy.phy.duke.edu.
220 DUKPHY.PHY.DUKE.EDU MultiNet FTP Server Process 3.3(14) at Mon 13-Jun-94
Name: yourname
331 User name (yourname) ok. Password, please.
Password:
ftp> binary
200 Type I ok.
ftp> quote site rms recsize 32400
200 IMAGE file record size now 32400 bytes
ftp> put myfile.rzhist myfile.rzhist
ftp> quit
```

12.2.2 Unpacking the BACKUP savesets

Having uncompressed and "resized" the BACKUP savesets, use the BACKUP command to unpack the files into the appropriate directories on your system.

Unpacking the BACKUP savesets

```
back/log disk$scratch:[pubmf.work.jamie]crnexe.bck/save disk$cernlib:[*...]
%BACKUP-S-CREDIR, created directory DISK$CD:[PRO.EXE]
%BACKUP-S-CREATED, created DISK$CD:[PRO.EXE]ARIADNE.EXE;12
%BACKUP-S-CREATED, created DISK$CD:[PRO.EXE]ASTUCE.EXE;53

...
```

12.3 Retrieving tar files for VM/CMS systems

Chapter 13: Retrieving individual source files

As the compressed tar files are only produced following a release of the complete libraries, it may be necessary to retrieve individual source files, e.g. if one wishes to install a new version of a library or modules. In this case, one should transfer the appropriate **.car** and **.cra** files.

N.B. please also check ensure that you copy the latest versions of any files in the *mgr* tree.

Retrieving individual source files

```
zfatal:/home/cp/jamie/hepdb (10) cd /tmp
zfatal:/tmp (11) ftp asisftp.cern.ch
Connected to asisftp.cern.ch.
220 asisftp FTP server (Version 2.0WU(14) Fri Sep 17 15:39:37 MET DST 1993) ready
.
Name (asisftp.cern.ch:jamie): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230-                  ----------------------------------------
230-                  Application Software Installation Server
230-                  ----------------------------------------
230-
230-   Welcome to the ASIS ftp server, developed by the CERN Computing and
230-   Networking Division to serve the High Energy Physics research community.
230-
230-   ftp clients may abort due to improper handling of such introductory
230-   messages. A dash (-) as the first character of your pw will suppress it.
230-
230-   The CERNlib software, located in the "cernlib" directory, is covered by
230-   CERN copyright. Before taking any material from this directory, please
230-   read the copyright notice "cernlib/copyright".
230-
230-   Please contact cernlib@cernvm.cern.ch for site registration. General
230-   support questions should be addressed to asis-support@asis01.cern.ch.
230-
230 Guest login ok, access restrictions apply.
ftp>
ftp> cd cernlib/share/94a/src/car
250---------------------------------------------------------------------------
250-     CERNlib release 94a:    scheduled date      March 1994
250-
250-     directory /cern/94a/src/car     containing Patchy sources *.car
250-                                                Patchy cradles *.cra
250-
250-                                     compressed Patchy sources *.zcar
250---------------------------------------------------------------------------
250-
250 CWD command successful.
ftp>
ftp> dir paw.car
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
-rw-r--r--   1 cernlib  software  4527809 Feb  1 10:47 paw.car
226 Transfer complete.
ftp> get paw.car
200 PORT command successful.
```

```
150 Opening ASCII mode data connection for paw.car (4527809 bytes).
226 Transfer complete.
4648428 bytes received in 44.68 seconds (101.6 Kbytes/s)
ftp>
```

N.B. as many of the CERN library packages depend on each other, you may require new versions of other packages as well.

Chapter 14: Overview of the installation procedures

The CERNLIB installation procedures perform the following steps:

- Extract the appropriate source code from the master file, e.g. generate the VMS or Unix specific version.
- Split the extracted code into separate files (typically on Unix machines).
- Compile the code
- Generate the libraries

In some cases, additional steps are required, e.g. processing of the **KUIP** *Command Definition File* by the KUIP compiler, or the assembly of individual components into a larger library (e.g. PACKLIB is composed of HBOOK [2], KUIP [3], ZEBRA [4] etc.). Currently, source code is typically extracted using the program **YPATCHY**. If splitting is required, this is performed by **FCASPLIT**. The basic functionality of these two programs is described below.

14.1 YPATCHY

The complete functionality of PATCHY is described in the PATCHY reference manual [5]. An introductory guide is given in the Patchy for beginners guide [6].

Rather than attempt to describe all of the features of PATCHY here, we will take the specific example of the HBOOK package.

To install HBOOK, two files are required. These are the HBOOK source file, typically kept in `/cern/pro/src/car/hbook.car` on Unix systems at CERN, and the so-called *cradle*.

The cradle for HBOOK, in `/cern/pro/src/car/hbook.cra`, is as follows.

Cradle for the installation of HBOOK

```
+EXE.
+USE,*HBOOK,$PLINAME.
+ASM,24.
+USE,QXNO_SC ,T=I,IF=QX_SC.
+USE,QX_SC   ,T=I,IF=QXNO_SC.
+USE,QXCAPT  ,T=I,IF=QXNO_SC,QX_SC.
+PAM,11,T=C,A.$CERN_ROOT/src/car/hbook
+QUIT.
```

Other cradles may be more complicated, but this will help describe the basic ideas. Let us examine each line of this cradle in turn.

1. +EXE.

   This tells YPATCHY to write out all 'material', typically source code, that has been selected. The material is written to the so-called Assembled Material file, or **ASM** file for short. Different streams exist for various types of material, as described below. They are all initially connected to the default stream on unit 21.

2. +USE,*HBOOK,$PLINAME.

   This, and other +USE statements, select the material of interest. Multi-level selection is possible. For example, *HBOOK will trigger all of the things in the hbook.car file in the PATCH *HBOOK.

   $PLINAME is set by the installation procedures. Typically, it is the machine type, e.g. DECS for DECstation, IBMRT for RS6000 etc.

   By convention, an asterix is used to indicate a so-called **pilot** patch, which will contain other **+USE** statements. The flags selected by $PLINAME are normally used to select machine specific features, as shown below.

   Flagging machine specific features using PATCHY

```
+SELF,IF=IBMRT.
*
*     RS 6000 specific code
*
+SELF.
```

   The same effect can be achieved using the C preprocessor, as is shown below.

   Flagging machine specific features with C preprocessor statements

```
#ifdef IBMRT

/* RS 6000 specific code
 */

#endif /* IBMRT */
```

3. +ASM,24.

   This tells PATCHY to establish a new output stream. By default, all material will be written to the stream 21, which is automatically initialised and does not require a +ASM directive to establish it. There are rules/conventions as to which streams are used for what:

21	Fortran
22	assembler
23	data
24	c
31	diverted Fortran

   ...

   The diverted streams are useful when different compilation options are required, e.g. static or noopt etc.

4. The next 3 +USE statements are to select the right sort of external names, typically for Fortran called C routines.

   The convention adopted is

`QX_SC`	external names are postfixed with an underscore, e.g. **hbook1_**. Most Unix systems append an underscore to external names in Fortran routines. On some systems, such as HP/UX and IBM RS6000, one must explicitly request this at compile time [1]. The trailing underscore is typically used to avoid name clashes between C and Fortran run-time libraries.
`QX_NOSC`	no underscore
`QX_CAPT`	no underscore and uppercase

   Examples of styles of external names

```
+SELF,IF=QXCAPT.
int CDHSTC(hnf)
+SELF,IF=QXNO_SC.
int cdhstc(hnf)
+SELF,IF=QX_SC.
int cdhstc_(hnf)
+SELF.
char *hnf;
```

5. +PAM,11,T=C,A.$CERN_ROOT/src/car/hbook

   This directive tells PATCHY to read the 'card' format file which contains the HBOOK source.

6. +QUIT.

   All done.

[1] The options for these two systems are **+ppu** and **-qextname** respectively.

Chapter 15: Installing PATCHY

Prior to installing the CERNLIB software from scratch, you must install **PATCHY**. You may obtain the **PATCHY** installation kit from as shown below.

15.1 Unix systems

15.1.1 Retrieving the binaries from asisftp.cern.ch

In most cases, the **PATCHY** binaries can be retrieved from asisftp.cern.ch as shown below. It may be necessary to rebuild the modules if there are compiler/shared library incompatibilities etc.

Retrieving the modules from asis

```
zfatal:/home/cp/jamie (4) ftp asisftp.cern.ch
Connected to asisftp.cern.ch.
220 asisftp FTP server (Version 2.0WU(14) Fri Sep 17 15:39:37 MET DST 1993) ready
.
Name (asisftp.cern.ch:jamie): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230-                  ----------------------------------------
230-                  Application Software Installation Server
230-                  ----------------------------------------
230-
230-   Welcome to the ASIS ftp server, developed by the CERN Computing and
230-   Networking Division to serve the High Energy Physics research community.
230-
230-   ftp clients may abort due to improper handling of such introductory
230-   messages. A dash (-) as the first character of your pw will suppress it.
230-
230-   The CERNlib software, located in the "cernlib" directory, is covered by
230-   CERN copyright. Before taking any material from this directory, please
230-   read the copyright notice "cernlib/copyright".
230-
230-   Please contact cernlib@cernvm.cern.ch for site registration. General
230-   support questions should be addressed to asis-support@asis01.cern.ch.
230-
230 Guest login ok, access restrictions apply.
ftp> cd cernlib/rs_aix32/patchy/4.15/bin
250 CWD command successful.
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
fcasplit
ycompar
yedit
yfrceta
yindex
yindexb
ylist
ylistb
ypatchy
ysearch
```

```
yshift
ytobcd
ytobin
ytoceta
226 Transfer complete.
ftp>
ftp>
Local directory now /home/cp/jamie/patchy/bin
ftp> bin
200 Type set to I.
ftp> prompt off
Interactive mode off.
ftp> mget *
200 PORT command successful.
150 Opening BINARY mode data connection for fcasplit (20539 bytes).
226 Transfer complete.
20539 bytes received in 0.8578 seconds (23.38 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for ycompar (73414 bytes).
226 Transfer complete.
73414 bytes received in 1.1 seconds (65.17 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for yedit (102039 bytes).
226 Transfer complete.
102039 bytes received in 0.6132 seconds (162.5 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for yfrceta (96265 bytes).
226 Transfer complete.
96265 bytes received in 0.7846 seconds (119.8 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for yindex (1299 bytes).
226 Transfer complete.
1299 bytes received in 0.05103 seconds (24.86 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for yindexb (79914 bytes).
226 Transfer complete.
79914 bytes received in 0.9767 seconds (79.91 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for ylist (1294 bytes).
226 Transfer complete.
1294 bytes received in 0.02697 seconds (46.85 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for ylistb (78498 bytes).
226 Transfer complete.
78498 bytes received in 0.9826 seconds (78.02 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for ypatchy (161238 bytes).
226 Transfer complete.
161238 bytes received in 0.9318 seconds (169 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for ysearch (88551 bytes).
226 Transfer complete.
88551 bytes received in 0.9307 seconds (92.91 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for yshift (93915 bytes).
226 Transfer complete.
93915 bytes received in 1.032 seconds (88.9 Kbytes/s)
```

```
200 PORT command successful.
150 Opening BINARY mode data connection for ytobcd (73968 bytes).
226 Transfer complete.
73968 bytes received in 0.4764 seconds (151.6 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for ytobin (84577 bytes).
226 Transfer complete.
84577 bytes received in 0.533 seconds (155 Kbytes/s)
200 PORT command successful.
150 Opening BINARY mode data connection for ytoceta (88839 bytes).
226 Transfer complete.
88839 bytes received in 0.4553 seconds (190.5 Kbytes/s)
ftp> quit
221 Goodbye.
zfatal:/home/cp/jamie (5)
```

15.1.2 Installing PATCHY from the installation kit

To install PATCHY from the installation kit, first retrieve the required files from asisftp.cern.ch as shown below.

Retrieving the PATCHY installation kit for Unix systems

```
zfatal:/home/cr/cernlib (415) ftp asisftp
Connected to asisftp.cern.ch.
220 asisftp FTP server (Version 2.0WU(14) Fri Sep 17 15:39:37 MET DST 1993) ready
.
Name (asisftp:cernlib): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230-                  ----------------------------------------
230-                  Application Software Installation Server
230-                  ----------------------------------------
230-
230-   Welcome to the ASIS ftp server, developed by the CERN Computing and
230-   Networking Division to serve the High Energy Physics research community.
230-
230-   ftp clients may abort due to improper handling of such introductory
230-   messages. A dash (-) as the first character of your pw will suppress it.
230-
230-   The CERNlib software, located in the "cernlib" directory, is covered by
230-   CERN copyright. Before taking any material from this directory, please
230-   read the copyright notice "cernlib/copyright".
230-
230-   Please contact cernlib@cernvm.cern.ch for site registration. General
230-   support questions should be addressed to asis-support@asis01.cern.ch.
230-
230 Guest login ok, access restrictions apply.
ftp>
ftp> cd cernlib/rs_aix32/patchy/4.15/src
250-Please read the file README
250-  it was last modified on Tue Nov 30 13:19:59 1993 - 63 days ago
250 CWD command successful.
```

```
ftp>
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
README
make_patchy
p4boot.sh0
p4inceta
rceta.sh
226 Transfer complete.
ftp>
ftp> get README
200 PORT command successful.
150 Opening ASCII mode data connection for README (4844 bytes).
226 Transfer complete.
4963 bytes received in 0.03007 seconds (161.2 Kbytes/s)
ftp> get make_patchy
200 PORT command successful.
150 Opening ASCII mode data connection for make_patchy (4434 bytes).
226 Transfer complete.
4548 bytes received in 0.02672 seconds (166.2 Kbytes/s)
ftp> get p4boot.sh0
200 PORT command successful.
150 Opening ASCII mode data connection for p4boot.sh0 (12888 bytes).
226 Transfer complete.
13335 bytes received in 0.1023 seconds (127.3 Kbytes/s)
ftp> get rceta.sh
200 PORT command successful.
150 Opening ASCII mode data connection for rceta.sh (8598 bytes).
226 Transfer complete.
8877 bytes received in 0.03805 seconds (227.8 Kbytes/s)
ftp> bin
200 Type set to I.
ftp> get p4inceta
200 PORT command successful.
150 Opening BINARY mode data connection for p4inceta (1573200 bytes).
226 Transfer complete.
1573200 bytes received in 8.896 seconds (172.7 Kbytes/s)
ftp>
ftp> quit
221 Goodbye.
zfatal:/home/cr/cernlib (416)
```

We now ensure that the variable **CERN** is correctly defined and then run **make_patchy**.

Running make_patchy

```
chmod +x make_patchy

export CERN=/cernlib/cern

./make_patchy
```

This will then launch the installation and verification procedure, resulting in the following files:

```
fcasplit
ycompar
yedit
yfrceta
yindex
ylist
ypatchy
ysearch
yshift
ytobcd
ytobin
ytoceta
```

Only **fcasplit** and **ypatchy** are required for the CERNLIB installation procedures.

15.2 VMS systems

15.2.1 Copying the PATCHY executables from VXCERN

The PATCHY modules may be copied from VXCERN as shown below.

Copying the PATCHY modules from VXCERN

```
COPY VXCERN::CERNVAX:[PATCHY.PRO.EXE]*.* * ! VAX versions

COPY VXCERN::CERNAXP:[PATCHY.PRO.EXE]*.* * ! AXP (Alpha) versions
```

15.2.2 Rebuilding PATCHY from the installation kit

On VAX/VMS systems, the installation kit is available on VXCERN as shown below.

N.B. an installation kit is only available for VAX systems, i.e. there is no Alpha installation kit.

Copying the PATCHY installation kit from VXCERN

```
VSCLIB? set def [.patchy]
VSCLIB? copy vxcern::cernvax:[patchy.src.vaxvms]*.* */log

%COPY-S-COPIED, VXCERN::CERNVAX:[PATCHY.SRC.VAXVMS]P4BOOT.SH0;2 copied to
 DISK$USER1:[JAMIE.PATCHY]P4BOOT.SH0;2 (23 blocks)
%COPY-S-COPIED, VXCERN::CERNVAX:[PATCHY.SRC.VAXVMS]P4INCETA.CET;8 copied to
 DISK$USER1:[JAMIE.PATCHY]P4INCETA.CET;8 (3235 blocks)
%COPY-S-COPIED, VXCERN::CERNVAX:[PATCHY.SRC.VAXVMS]PATCHY.COM;4 copied to
 DISK$USER1:[JAMIE.PATCHY]PATCHY.COM;4 (3 blocks)
```

```
%COPY-S-COPIED, VXCERN::CERNVAX:[PATCHY.SRC.VAXVMS]RCETA.SH;3 copied to
 DISK$USER1:[JAMIE.PATCHY]RCETA.SH;3 (16 blocks)
%COPY-S-COPIED, VXCERN::CERNVAX:[PATCHY.SRC.VAXVMS]README.DOC;3 copied to
 DISK$USER1:[JAMIE.PATCHY]README.DOC;3 (10 blocks)
%COPY-S-NEWFILES, 5 files created
```

Now customise **PATCHY.COM** specifying the source, work and target directories. The modules are then built by typing **@PATCHY**.

Result of running PATCHY.COM

```
Directory DISK$USER1:[JAMIE.PATCHY.EXE]

YCOMPAR.EXE;1       YEDIT.EXE;1         YFRCETA.EXE;1       YINDEX.EXE;1
YLIST.EXE;1         YPATCHY.EXE;1       YSEARCH.EXE;1       YSHIFT.EXE;1
YTOBCD.EXE;1        YTOBIN.EXE;1        YTOCETA.EXE;1

Total of 11 files.
```

15.3 VM systems

15.4 Other systems

Chapter 16: Installing CERNLIB software on Unix systems

Below we describe two scenarios. The first is for a Unix system at CERN, where the asis tree is available via **NFS** or **AFS**. The second is for a remote system or for one where the asis tree is not available.

16.1 Installing CERNLIB when asis is available

In the following examples, the CERNLIB tree is available via AFS. The procedure is identical for the case when the CERNLIB tree is mounted via **NFS**.

Accessing the CERNLIB tree via AFS

```
zfatal:/hepdb/cdchorus (185) ls -l /cern
total 0
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 93c -> /afs/cern.ch/asis/cern/93c
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 93d -> /afs/cern.ch/asis/cern/93d
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 94a -> /afs/cern.ch/asis/cern/94a
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 WWW -> /afs/cern.ch/asis/cern/WWW
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 cmz -> /afs/cern.ch/asis/cern/cmz
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 mad -> /afs/cern.ch/asis/cern/mad
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 man -> /afs/cern.ch/asis/cern/man
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 new -> /afs/cern.ch/asis/cern/new
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 old -> /afs/cern.ch/asis/cern/old
lrwxrwxrwx   1 root     system        29 Dec  7 21:02 patchy -> /afs/cern.ch/asis/cern/patchy
lrwxrwxrwx   1 root     system        28 Dec  7 21:02 phigs -> /afs/cern.ch/asis/cern/phigs
lrwxrwxrwx   1 root     system        26 Dec  7 21:02 pro -> /afs/cern.ch/asis/cern/pro
lrwxrwxrwx   1 root     system        28 Dec  7 21:02 share -> /afs/cern.ch/asis/cern/share
```

Let us assume that we wish to reinstall the CERNLIB software in the **/cernlib/cern** tree. We first create these directories, and then a subdirectory for the version that we wish to install. We procede as follows:

Setting up the directory tree

```
mkdir /cernlib/cern
mkdir /cernlib/cern/93d
mkdir /cernlib/cern/93d/bin
mkdir /cernlib/cern/93d/lib
mkdir /cernlib/cern/93d/log
mkdir /cernlib/cern/93d/src
mkdir /cernlib/cern/93d/doc
```

We now set up a number of links.

Creating links into the AFS tree

```
cd /cernlib/cern/93d
ln -s /cern/93d/include include
ln -s /cern/93d/mgr mgr
cd src
ln -s /cern/93d/src/car car
ln -s /cern/93d/src/cmz cmz
ln -s car cra
```

In fact, only the links for the **car** and **cra** directories are required for what follows.

`car`	The CERNLIB source files in **PATCHY** card format. This directory also contains the **PATCHY** *cradles* (*.cra) used to extract the code.
`cmz`	The CERNLIB source files in **CMZ** binary format.
`cra`	A link to the **cra** directory.
`doc`	Documentation on various Monte Carlo generators.
`include`	*include* files used when the CERNLIB code is called from C.

We now add the following commands to our profile.

Tailoring the .profile of the cernlib account

```
PATH=/cern/pro/bin:$PATH; export PATH

export CERN=/cernlib/cern
export CERN_LEVEL=93d
export PLISTA=DEV
. $CERN/$CERN_LEVEL/mgr/plienv.sh
```

We then reexecute the **.profile** and switch to the CERN manager directory.

Preparing to build the CERN software

```
. .profile

cd $CERN/$CERN_LEVEL/mgr
```

We can now build the complete CERN software by typing **make all**.

Building the CERN software

```
   make -n all

   ...

makepack -p kerngen
makepack -s -c kerngen

   ...

makepack -s -c kernasw
makepack -l kernlib
makepack -p cspack
makepack -s cspack
makepack -l packlib -c cspack

   ...

makepack -p isajetd
rm -r /cernlib/cern/93d/src/cfs/isajetd
makepack -p pdflibd
rm -r /cernlib/cern/93d/src/cfs/pdflibd
```

Various components can be built using the syntax **make** *target*. Thus, to build the PAW modules one would type **make paw**. As the standard Unix **make** is employed, all the dependancies are known and intermediate components only rebuilt if required.

The following extract from the **makefile** indicates which components can be rebuilt separately or together.

Extract from cernlib makefile

```
# **********************************************************************
# Make definitions                                                     *
# **********************************************************************
# ======================================================================
# >>> General makes
# ======================================================================
# all:        cernset products
  all:        cernset
  cernset:    cernlibs cernpgm userpgm mclibs mcdoc

  cernlibs:   kernlib packlib mathlib graflibs pawlib phtools
  cernpgm:    dzedit fatset kuipset paw rzconv flop tree telnetg

              zftp pawserv zserv higzconv f2h hepdbset umlog
  userpgm:    garfield poisson
  mclibs:     ariadne cojets eurodec fritiof herwig isajet

              jetset lepto pdflib photos
  mcdoc:      cojetsd eurodecd fritiofd herwigd isajetd jetsetd

              pdflibd photosd pythiad
  shrlibs:    scernlib smathlib sgraflib sgeant
```

```
  products:   cmz gks historian nag
# =====================================================================
# >>> Basic Libraries
# =====================================================================
  kernlib:    kernlib.a
  packlib:    packlib.a
  mathlib:    mathlib.a
  phtools:    phtools.a

  graflibs:   graflib grafX11 grafGKS
  graflib:    graflib.a
  grafX11:    grafX11.a
  grafGKS:    grafGKS.a
  grafDGKS:   grafDGKS.a
  grafGL:     grafGL.a
  grafGPR:    grafGPR.a
  pawlib:     pawlib.a

  scernlib:   scernlib.a
  smathlib:   smathlib.a
  sgraflib:   sgraflib.a

  kernlib.a:  $(LIB)/libkernlib.a
  packlib.a:  $(LIB)/libpacklib.a
  mathlib.a:  $(LIB)/libmathlib.a
  phtools.a:  $(LIB)/libphtools.a

  graflib.a:  $(LIB)/libgraflib.a
  grafX11.a:  $(LIB)/libgrafX11.a
  grafGKS.a:  $(LIB)/libgrafGKS.a
  grafDGKS.a: $(LIB)/libgrafDGKS.a
  grafGL.a:   $(LIB)/libgrafGL.a
  grafGPR.a:  $(LIB)/libgrafGPR.a
  pawlib.a:   $(LIB)/libpawlib.a

  scernlib.a: $(LIB)/scernlib.a
  smathlib.a: $(LIB)/smathlib.a
  sgraflib.a: $(LIB)/sgraflib.a
```

16.2 Installing CERNLIB without asis

If the CERN library directory tree is not accessible over **NFS** or **AFS**, we must first retrieve the compressed tar files containing the source (cernsrc.tar.Z) and the installation scripts (cernmgr.tar.Z). [1]

We first connect to the asis server, as shown below.

Retrieving the CERNLIB sources and installation scripts

```
zfatal:/cernlib/tmp (132) ftp asisftp.cern.ch
Connected to asisftp.cern.ch.
220 asisftp FTP server (Version 2.0WU(14) Fri Sep 17 15:39:37 MET DST 1993) ready
```

[1] Note that files are now packed using **GZIP** and hence have extension .gz.

```
.
Name (asisftp.cern.ch:cernlib): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230-                   ---------------------------------------
230-                   Application Software Installation Server
230-                   ---------------------------------------
230-
230-   Welcome to the ASIS ftp server, developed by the CERN Computing and
230-   Networking Division to serve the High Energy Physics research community.
230-
230-   ftp clients may abort due to improper handling of such introductory
230-   messages. A dash (-) as the first character of your pw will suppress it.
230-
230-   The CERNlib software, located in the "cernlib" directory, is covered by
230-   CERN copyright. Before taking any material from this directory, please
230-   read the copyright notice "cernlib/copyright".
230-
230-
230-   Please contact cernlib@cernvm.cern.ch for site registration. General
230-   support questions should be addressed to asis-support@asis01.cern.ch.
230-
230 Guest login ok, access restrictions apply.
ftp>cd cernlib/rs_aix32/93d/tar
250-
250- This directory contains compressed files of CERNlib release 93d for IBM/RS6
000
250-
250- Files ending in .tar.gz have been compressed using gzip. gzip/gunzip for
250- IBM/RS6000 are available in this directory in the gzip.tar file. Get this
250- first and untar in a directory in the search path. Also take a new copy of

250- plitar; this will use gzip -d ( equivalent to gunzip ) to uncompress files
250- ending in .tar.gz.
250-
250-
250 CWD command successful.
ftp>

ftp> get plitar
200 PORT command successful.
150 Opening ASCII mode data connection for plitar (10951 bytes).
226 Transfer complete.
11262 bytes received in 0.06656 seconds (165.2 Kbytes/s)
ftp> get cernsrc.contents
200 PORT command successful.
150 Opening ASCII mode data connection for cernsrc.contents (22467 bytes).
226 Transfer complete.
22799 bytes received in 0.2305 seconds (96.6 Kbytes/s)
ftp> get cernmgr.contents
200 PORT command successful.
150 Opening ASCII mode data connection for cernmgr.contents (4937 bytes).
226 Transfer complete.
5016 bytes received in 0.01622 seconds (302 Kbytes/s)
ftp> bin

ftp> get cernmgr.tar.Z
```

```
200 PORT command successful.
150 Opening BINARY mode data connection for cernmgr.tar.Z (111289 bytes).
226 Transfer complete.
111289 bytes received in 0.7157 seconds (151.9 Kbytes/s)
ftp> get cernsrc.tar.Z
200 PORT command successful.
150 Opening BINARY mode data connection for cernsrc.tar.Z (19319023 bytes).
226 Transfer complete.
19319023 bytes received in 118.1 seconds (159.8 Kbytes/s)
ftp>
```

Having retrieved the installation kits, we now unpack using **plitar**.

Unpacking the installation procedures

```
zfatal:/cernlib/tmp (133) ls -l
total 38048
-rw-r--r--   1 cernlib  sys         4937 Feb  1 14:05 cernmgr.contents
-rw-r--r--   1 cernlib  sys       111289 Feb  1 14:06 cernmgr.tar.Z
-rw-r--r--   1 cernlib  sys        22467 Feb  1 14:05 cernsrc.contents
-rw-r--r--   1 cernlib  sys      19319023 Feb  1 14:08 cernsrc.tar.Z
-rw-r--r--   1 cernlib  sys        10951 Feb  1 14:05 plitar

zfatal:/cernlib/tmp (134) chmod +x plitar  # Make 'plitar' executable
zfatal:/cernlib/tmp (135)
zfatal:/cernlib/tmp (135) plitar           # See if it gives us any help
##====================================================================
##
## PLITAR   93d.02  : CERN Program Library distribution utility
## Last update      : 93/10/28
##
##====================================================================
#
#  Syntax: plitar [ -n ] tar_options tar_file
#
#  plitar combines tar+compress utilities to pack/unpack the files being
#  part of the CERNlib distribution set; the corresponding _readme file
#  describes the contents of each of them. Please read it beforehand.
#  Location of tar files and the target install directory is controlled
#  through environment variables CERN, CERN_LEVEL and PLITMP
#
#  Examples:
#
#  plitar -n xvf cernlib   "non-execute" mode to display the action
#  plitar tvf cmz          examines the CMZ compressed-tar set
#  plitar xvf geant        installs the GEANT compressed-tar set
#
zfatal:/cernlib/tmp (136) echo $CERN
/cernlib/cern
zfatal:/cernlib/tmp (137) export CERN=/cernlib/kern
zfatal:/cernlib/tmp (138) mkdir $CERN
zfatal:/cernlib/tmp (139) echo $CERN_LEVEL
93d
zfatal:/cernlib/tmp (140) echo $PLITMP
```

```
/tmp
zfatal:/cernlib/tmp (141) export PLITMP=/cernlib/tmp
zfatal:/cernlib/tmp (145) plitar -n tvf cernmgr.tar.Z
##=================================================================
##
## PLITAR   93d.02  : CERN Program Library distribution utility
## Last update      : 93/10/28
##
##=================================================================
#
# -------------------------------------------------------------------
# The 2 parameters CERN and PLITMP are environment variables
# which may be changed using setenv (in C-shell) or export (in sh,ksh)
# -------------------------------------------------------------------
#
# Tar files expected in      PLITMP=/cernlib/tmp
# Target directory           CERN  =/cernlib/kern
#
         uncompress -vc /cernlib/tmp/cernmgr.tar.Z |  tar tvf  -


zfatal:/cernlib/tmp (146) plitar xvf cernmgr.tar.Z
##=================================================================
##
## PLITAR   93d.02  : CERN Program Library distribution utility
## Last update      : 93/10/28
##
##=================================================================
#
# -------------------------------------------------------------------
# The 2 parameters CERN and PLITMP are environment variables
# which may be changed using setenv (in C-shell) or export (in sh,ksh)
# -------------------------------------------------------------------
#
# Tar files expected in      PLITMP=/cernlib/tmp
# Target directory           CERN  =/cernlib/kern
#
         uncompress -vc /cernlib/tmp/cernmgr.tar.Z |  tar xvf  -
x 93d/mgr/93d/irs.names, 4679 bytes, 10 media blocks.
x 93d/mgr/93d/dos.names, 3548 bytes, 7 media blocks.

...

x 93d/mgr/yexpand, 1230 bytes, 3 media blocks.
```

We can now unpack the source files in the same way.

Unpacking the source files

```
zfatal:/cernlib/tmp (150) plitar xvf cernsrc.tar.Z
##=================================================================
##
## PLITAR   93d.02  : CERN Program Library distribution utility
## Last update      : 93/10/28
```

```
##
##====================================================================
#
# ---------------------------------------------------------------------
# The 2 parameters CERN and PLITMP are environment variables
# which may be changed using setenv (in C-shell) or export (in sh,ksh)
# ---------------------------------------------------------------------
#
# Tar files expected in      PLITMP=/cernlib/tmp
# Target directory           CERN  =/cernlib/kern
#
         uncompress -vc /cernlib/tmp/cernsrc.tar.Z |  tar xvf  -
x 93d/src/car/comis.cra, 393 bytes, 1 media blocks.
x 93d/src/car/wylbur.car, 1063629 bytes, 2078 media blocks.
x 93d/src/car/kernvax.car, 196447 bytes, 384 media blocks.
x 93d/src/car/ariadne.cra, 134 bytes, 1 media blocks.

...

x 93d/src/car/pawdemo.car, 784187 bytes, 1532 media blocks.
x 93d/src/car/kernsgi2.car is a symbolic link to kernsgi.car.
x 93d/src/car/isajet72.car, 1226674 bytes, 2396 media blocks.
x 93d/src/car/isajet72.cra, 152 bytes, 1 media blocks.
zfatal:/cernlib/tmp (151)
```

We now add the following commands to our profile.

Tailoring the .profile of the cernlib account

```
PATH=/cern/pro/bin:$PATH; export PATH

export CERN=/cernlib/cern
export CERN_LEVEL=93d
export PLISTA=DEV
. $CERN/$CERN_LEVEL/mgr/plienv.sh
```

We then reexecute the **.profile** and switch to the CERN manager directory.

Preparing to build the CERN software

```
. .profile

cd $CERN/$CERN_LEVEL/mgr
```

We now create 3 directories and 1 link and are then ready to start the installation.

Completing the pre-installation phase

```
zfatal:/cernlib/tmp (160) cd $CERN/$CERN_LEVEL
zfatal:/cernlib/kern/93d (161) mkdir bin
zfatal:/cernlib/kern/93d (162) mkdir lib
zfatal:/cernlib/kern/93d (163) mkdir log
zfatal:/cernlib/kern/93d (164) mkdir doc
zfatal:/cernlib/kern/93d (165)

zfatal:/cernlib/kern/93d (166) cd src
zfatal:/cernlib/kern/93d/src (165) ln -sf car cra
zfatal:/cernlib/kern/93d/src (166)
```

N.B. the source code of the package *MPA* is not available for distribution. For make all to work, the dummy link *mpa.car* in the *src/car* directory must be removed.

Removing dummy mpa.car file

```
rm /cernlib/kern/93d/src/car/mpa.car # as root if all else fails
touch /cernlib/kern/93d/src/car/mpa.car
```

We may now rebuild the entire CERN library using **make all** or one component, as shown below.

Building the FATMEN module

```
zfatal:/cernlib/kern/93d/mgr (181) make -n fatmen
        [ "n" = "n" -o "n" = "bn" ] && make -n -f userlib -f grouplib -f /cernlib/kern/93d/mgr/mkf
/cernlib  -f /cernlib/kern/93d/mgr/mkf/geant fatmen

                            || make -n -f userlib -f grouplib -f /cernlib/kern/93d/mgr/mkf/cernlib  -f /cernlib/kern/93d/mgr/mkf/geant fatmen >>/cernlib/kern/93d/log/fatmen 2>&1
        makepack -p kuipc
        makepack -s -c kuipc
        makepack -o kuipc kuipc
        makepack -p fatmen
        makepack -s -c fatmen
        makepack -p cspack
        makepack -s cspack
        makepack -l packlib -c cspack
        makepack -p cdlib
        makepack -s cdlib
        makepack -l packlib -c cdlib
        makepack -p epio
        makepack -s epio
        makepack -l packlib -c epio
        makepack -p fatlib
        makepack -s fatlib
        makepack -l packlib -c fatlib
        makepack -p ffread
        makepack -s ffread
        makepack -l packlib -c ffread
```

```
makepack -p hbook
makepack -s hbook
makepack -l packlib -c hbook
makepack -p kapack
makepack -s kapack
makepack -l packlib -c kapack
makepack -p kuip
makepack -s kuip
makepack -l packlib -c kuip
makepack -p minuit
makepack -s minuit
makepack -l packlib -c minuit
makepack -p zbook
makepack -s zbook
makepack -l packlib -c zbook
makepack -p zebra
makepack -s zebra
makepack -l packlib -c zebra
makepack -p kerngen
makepack -s -c kerngen
makepack -p kernnum
makepack -s -c kernnum
makepack -p kernasw
makepack -s -c kernasw
makepack -l kernlib
makepack -l packlib
makepack -o fatmen fatmen
```

Chapter 17: Installing CERNLIB software on VMS systems

CERNLIB software is installed on VMS systems using the **CERN_ROOT:[MGR]MAKE.COM** procedure. The symbol **MAKE** is defined as **@CERN_ROOT:[MGR]MAKE** by the procedure **PLIENV**, described on page 127. The following examples show how one may build library components, complete libraries or packages.

17.1 Building standalone libraries

Complete libraries may be built using the syntax **make** ***target***. For example, **KERNLIB** is built as follows:

Building KERNLIB

```
vxcrna:/cernlib > make -n kernlib
        makepack -p KERNASW
        makepack -s KERNASW
        makepack -c KERNASW
        makepack -p KERNNUM
        makepack -s KERNNUM
        makepack -c KERNNUM
        makepack -p KERNGEN
        makepack -s KERNGEN
        makepack -c KERNGEN
        makepack -l KERNLIB
```

As for the standard Unix make, the option **-n** tells **make** just to list what it would do and not actually execute the commands.

PACKLIB may be built in a similar manner, as shown below.

Building PACKLIB

```
vxcrna:/cernlib > make -n packlib
        makepack -p CSPACK
        makepack -s CSPACK
        makepack -c CSPACK
        makepack -p EPIO
        makepack -s EPIO
        makepack -c EPIO
        makepack -p FATLIB
        makepack -s FATLIB
        makepack -c FATLIB
        makepack -p FFREAD
        makepack -s FFREAD
        makepack -c FFREAD
        makepack -p HBOOK
        makepack -s HBOOK
        makepack -c HBOOK
        makepack -p KAPACK
        makepack -s KAPACK
```

```
        makepack -c KAPACK
        makepack -p KUIP
        makepack -s KUIP
        makepack -c KUIP
        makepack -p MINUIT
        makepack -s MINUIT
        makepack -c MINUIT
        makepack -p ZBOOK
        makepack -s ZBOOK
        makepack -c ZBOOK
        makepack -p ZEBRA
        makepack -s ZEBRA
        makepack -c ZEBRA
        makepack -p CDLIB
        makepack -s CDLIB
        makepack -c CDLIB
        makepack -l PACKLIB
```

Both KERNLIB and PACKLIB contain a number of components. Let us first examine how a library containing only one component is built.

17.2 Building a simple library

Building JETSET

```
vxcrna:/cernlib > make -n jetset
        makepack -p JETSET
        makepack -s JETSET
        makepack -c JETSET
        makepack -l JETSET
```

JETSET is built in four steps:

- `-p` This invokes the **PATCHY** step, to extract the source code from the **.CAR** file.
- `-s` This invokes the **FCASPLIT** step, to split the extracted code into separate files, one per routine (strictly, one per PATCHY DECK).
- `-c` This invokes the compile step, to compile the individual routines.
- `-l` This creates the library out of the object files created by the previous step.

17.3 Building a complex library

A complex library, such as KERNLIB or PACKLIB, may be built in one go, as shown above, or component by component. The former is useful when one wishes to install a new release of CERNLIB, or install CERNLIB from scratch. The latter is more appropriate if only one or a few packages have changed.

For example, if a routine in ZEBRA has been modified, we may rebuild PACKLIB using the following steps:

1. make zebra

2. makepack -l packlib

The first command will cause the ZEBRA source code to be extracted, split and compiled. The second will rebuild PACKLIB from all of the appropriate object files.

17.4 Building a module

Modules are built in a similar manner. For example, the HEPDB server **CDSERV** is built as follows:

Building the HEPDB server CDSERV

```
vxcrna:/cernlib > make -n cdserv
        makepack -p CDSERV
        makepack -c CDSERV
        makepack -o CDSERV  CDSERV
```

Here we see that three steps are involved.

- `-p` The source code of the server is extract by the PATCHY step.
- `-c` The code is compiled.
- `-o` The code is linked to produce an executable module.

17.5 Building sets of modules

One may also build multiple modules in one go. For example, rather than rebuild all different versions of PAW individually, one may request that they are all rebuilt, as follows:

Building multiple modules

```
vxcrna:/cernlib > make -n pawall
        makepack -p PAWMDNET
        makepack -s PAWMDNET
        makepack -c PAWMDNET
        makepack -o PAWX11   PAWMDNET
        makepack -p PAWMDNET
        makepack -s PAWMDNET
        makepack -c PAWMDNET
        makepack -o PAWDECW  PAWMDNET
        makepack -p PAWMDNET
        makepack -s PAWMDNET
        makepack -c PAWMDNET
        makepack -o PAWGKS  PAWMDNET
        makepack -p PAWPP
        makepack -s PAWPP
        makepack -c PAWPP
        makepack -o PAWPP    PAWPP
        makepack -p KXTERM
        makepack -s KXTERM
        makepack -c KXTERM
        makepack -o KXTERM   KXTERM
```

```
makepack -p PAWM
makepack -s PAWM
makepack -c PAWM
makepack -o PAWGKS_T PAWM
makepack -p PAWM
makepack -s PAWM
makepack -c PAWM
makepack -o PAWX11_T PAWM
```

In this case, KXTERM is also rebuilt as it is required by PAW++.

17.6 Handling dependencies

As we are not using a true *make* utility on VMS systems, the installer must be aware of the dependencies of various components of CERNLIB. This has the following consequences:

1. A complete installation of CERNLIB from scratch must be done in the correct order.
2. When reinstalling a particular package, one must take care to reinstall all components that have changed in the correct order.

We hope to introduce a *make* or make-like utility which will simplify the installation.

17.7 Recommended procedure for installing CERNLIB

N.B. if there has been no previous installation of CERNLIB on your system, see section 9.

1. Build the libraries in the following order:
   (a) **KERNLIB**
   (b) **MATHLIB**
   (c) **PACKLIB**
   (d) Graphics libraries
      - **GRAFLIB**
      - GRAFGKS
      - GRAFDGKS
      - GRAFX11
2. Make PAWLIB
3. Make the modules (PAW, FATMEN, HEPDB etc.)
4. Make the Monte Carlo and other stand alone libraries
   - ARIADNE
   - COJETS
   - EURODEC

- FRITIOF
- HERWIG
- ISAJET
- JETSET
- LEPTO
- PDFLIB
- PHOTOS
- PHTOOLS
- TWISTER

17.7.1 Rebuilding the complete CERN libraries

All of the libraries and modules can be rebuilt using the following command file.

MAKEALL.COM

```
$!
$! Make complete CERNLIB
$!
$ set noon
$ save_message = f$environment("MESSAGE")
$!
$       warnings_from = "."
$       errors_from   = "."
$       severe_from   = "."
$!
$ cernlib = -
"KERNLIB,MATHLIB,PACKLIB,GRAFLIB,GRAFGKS,GRAFDGKS,GRAFX11,PAWLIB" + -
",CERNPGM,USERPGM,MCLIBS,GEANT321"
$ if p1 .nes. "" then cernlib = p1
$ count   = 0
$ packages:
$ set message 'save_message'
$ package = f$element(count,",",cernlib)
$ if package .eqs. "," then goto end
$ write sys$output "Building ''package' at ''f$time()'"
$ make 'package'
$ wait 0:0:10
$ set message/nofacility/noidentification/noseverity/notext
$ p_wait:
$ show process/nooutput 'package'
$ if $severity .eq. 0
$ then
$    write sys$output "''package' complete at ''f$time()'"
$          search cern:[new.log]'package'.log "*** "
$!
$          search/nooutput cern:[new.log]'package'.log "*** WARNING EXIT from"
$    if $severity .eq. 1 then warnings_from = warnings_from + package + "."
$          search/nooutput cern:[new.log]'package'.log "*** ERROR EXIT from"
$    if $severity .eq. 1 then errors_from = errors_from + package + "."
$          search/nooutput cern:[new.log]'package'.log "*** SEVERE ERROR EXIT from"
$    if $severity .eq. 1 then severe_from = severe_from + package + "."
```

```
$!
$    count = count + 1
$    goto packages
$ endif
$ wait 0:1
$ goto p_wait
$ end:
$ write sys$output "CERNLIB build complete at ''f$time()'"
$ set message 'save_message'
$ if warnings_from .nes. "." then write sys$output -
"Warnings from ''warnings_from'"
$ if errors_from .nes.   "." then write sys$output -
"Errors from ''errors_from'"
$ if severe_from .nes.   "." then write sys$output -
"Severe errors from ''severe_from'"
```

17.7.2 Rebuilding PAW

All of the PAW modules, libraries and associated packages can be rebuilt using the following command file.

Command file to rebuild PAW

```
$!
$! Rebuild PAW
$!
$ set noon
$ save_message = f$environment("MESSAGE")
$!
$       warnings_from = "."
$       errors_from   = "."
$       severe_from   = "."
$!
$ cernlib = "PAWLIB,GRAFLIB,GRAFGKS,GRAFDGKS,GRAFX11,KUIP,PACKLIB,PAWALL"
$ if p1 .nes. "" then cernlib = p1
$ count   = 0
$ packages:
$ set message 'save_message'
$ package = f$element(count,",",cernlib)
$!
$! Treat PACKLIB specially
$!
$       if package .eqs. "PACKLIB"
$       then
$          makepack -l packlib
$          count = count + 1
$          goto packages
$       endif
$!
$ if package .eqs. "," then goto end
$ write sys$output "Building ''package' at ''f$time()'"
$ make 'package'
$ wait 0:0:10
$ set message/nofacility/noidentification/noseverity/notext
$ p_wait:
```

```
$ show process/nooutput 'package'
$ if $severity .eq. 0
$ then
$    write sys$output "''package' complete at ''f$time()'"
$          search cern:[new.log]'package'.log "*** "
$!
$          search/nooutput cern:[new.log]'package'.log "*** WARNING EXIT from"
$    if $severity .eq. 1 then warnings_from = warnings_from + package + "."
$          search/nooutput cern:[new.log]'package'.log "*** ERROR EXIT from"
$    if $severity .eq. 1 then errors_from = errors_from + package + "."
$          search/nooutput cern:[new.log]'package'.log "*** SEVERE ERROR EXIT from"
$    if $severity .eq. 1 then severe_from = severe_from + package + "."
$!
$    count = count + 1
$    goto packages
$ endif
$ wait 0:1
$ goto p_wait
$ end:
$ write sys$output "PAW build complete at ''f$time()'"
$ set message 'save_message'
$ if warnings_from .nes. "." then write sys$output -
"Warnings from ''warnings_from'"
$ if errors_from .nes.   "." then write sys$output -
"Errors from ''errors_from'"
$ if severe_from .nes.   "." then write sys$output -
"Severe errors from ''severe_from'"
```

17.7.3 Installing KERNLIB

KERNLIB can be built using the command **make kernlib**.

17.7.4 Installing MATHLIB

MATHLIB can be built using the command **make mathlib**. This will execute the following commands:

Building MATHLIB

```
vxcrna:/cernlib > make -n mathlib
        makepack -p LAPACK
        makepack -s LAPACK
        makepack -c LAPACK
        makepack -p BVSL
        makepack -s BVSL
        makepack -c BVSL
        makepack -p MPA
        makepack -s MPA
        makepack -c MPA
        makepack -p GEN
        makepack -s GEN
        makepack -c GEN
        makepack -l MATHLIB
```

17.7.5 Installing PACKLIB

PACKLIB can be built using the command `make packlib`. PACKLIB currently consists of the following packages:

- CSPACK [7]
- EPIO [8]
- FATMEN [9]
- FFREAD [10]
- HBOOK [2]
- HEPDB [11]
- KAPACK [12]
- KUIP [3]
- MINUIT [13]
- ZBOOK [14]
- ZEBRA [4]

17.7.6 Installing graphics libraries

The graphics libraries are divided into a kernel library, **GRAFLIB**, and package specific libraries:

`GRAFGKS`	GTS-GRAL GKS specific routines
`GRAFDGKS`	DEC GKS specific routines
`GRAFX11`	X11 specific routines

They may be installed using the command **make GRAFLIB GRAFGKS GRAFDGKS GRAFX11**.

17.7.7 Building the Monte Carlo and other stand alone libraries

These may be made using the command **make *target***. Note that some of the packages are available in several versions. For example, versions 6.3, 7.3 and 7.4 of **JETSET** are all available at the time of writing. The available versions can be found as shown below.

Listing the available versions of a given package

```
vxcrna:/cernlib > dir cern:[new.src.car]jetset*.car

Directory CERN:[NEW.SRC.CAR]

JETSET63.CAR;1      JETSET73.CAR;1      JETSET74.CAR;1

Total of 3 files.
```

Version 7.4 of **JETSET** is installed by typing **make jetset74**.
In a number of cases, the documentation is extracted by typing **make targetD**, e.g. **make herwig54D**. If this is the case, you will find an appropriate cradle in the **CERN_LEVEL:[SRC.CAR]** area, e.g. **herwig54D.cra**.

17.7.8 Installing PAWLIB

The PAW library **PAWLIB** is required if you wish to build PAW or if you intend to link your own applications with the PAW routines. It is built using the command **make pawlib**. This will extract and compile the various components, as shown below.

Building PAWLIB

```
vxcrna:/cernlib > make -n pawlib
        makepack -p PAW
        makepack -s PAW
        makepack -c PAW
        makepack -p COMIS
        makepack -s COMIS
        makepack -c COMIS
        makepack -p SIGMA
        makepack -s SIGMA
        makepack -c SIGMA
        makepack -l PAWLIB
```

17.7.9 Building the CERNLIB modules

All of the following modules can be built using the syntax **make *target*** except where indicated. Some modules can be built together, e.g. the various versions of PAW.

`AKMULT`	Program to split a file containing multiple Macro routines into individual files. (Obsolete?)
`HTONEW`	Program to convert HBOOK files. Not built with CERNLIB procedures.

- Sundry packages

`ASTUCE`	Extract source from Historian/Update file. Only supported for MAD.
`CERNLIB`	The CERNLIB command.
`FCONV`	Convert a file between different formats.
`FLOP`	Fortran coding convention checker.
`HIGZCONV`	Convert HIGZ RZ metafiles into postscript or GKS format.
`SYSREQ`	The SYSREQ command (used at CERN to access the Tape Management System)
`TREE`	Show calling tree of a Fortran program.
`TNX11_M`	TELNETG program linked with Multinet TCP/IP. See the CSPACK manual for details [7].
`WYLBUR`	A portable extended Wylbur like editor.
`XBANNER`	Program to write text in large letters.

- Zebra bank documentation programs. **make dzedit** builds DZEX11 and DZEGKS.

`DZDOC`	Zebra bank documentation package.
`DZEX11`	Interactive program linked with X11.
`DZEGKS`	As above, linked with GTS-GRAL GKS.

- FATMEN programs. Can be built individually or collectively using **make fatset**. See also the FATMEN manual [9] for information on configuring the FATMEN servers.

FATMEN	The FATMEN shell.
FATNEW	Program to build a new FATMEN catalogue.
FATSEND	Program to transfer FATMEN updates between servers.
FATSRV	The FATMEN server.

- Garfield programs.

GARFIELD
GARFRUN

- HEPDB programs. Can be built individually or collectively using **make hepdbset**. See also the HEPDB manual [11] for information on configuring the HEPDB servers.

HEPDB	The interactive interface.
CDSERV	The HEPDB server.
CDMAKE	Program to make a new database.
CDMOVE	Program to move updates between server queues.

- KUIP programs. Can be built individually or collectively using **make kuipset**.

KUIPC	The KUIP compiler.
KUESVR	The KUIP server.
KXTERM	The KUIP terminal used in applications such as **paw++**. Can also be built by typing **make paw++** or **make pawall**.

- The PAW [15] clients and server. Can be built individually (as shown) or collectively, with the exception of PAWSERV, via **make pawall**, which also builds **KXTERM**. **make pawset** can be used to make all PAW clients (i.e. not PAWSERV) except PAW++. **N.B. the modules linked with GTS-GRAL GKS are only available on VAX systems**.

PAWPP	PAW++
PAWSERV	The PAW server.
PAWX11	PAW linked with the X11 libraries, no TCP/IP.
PAWX11_DECW	PAW linked with the old DECWindows libraries, no TCP/IP.
PAWX11_M	PAW linked with the X11 libraries and Multinet TCP/IP.
PAWGKS	PAW linked with GTS-GRAL GKS.
PAWGKS_M	PAW linked with GTS-GRAL GKS and Multinet TCP/IP.

- Programs to permit transfer of RZ files. These may be built individually, e.g. **make rfra**, or collectively using **make rzconv**.

RFRA	Convert an RZ file from FZ alpha exchange format
RFRX	Convert an RZ file from FZ binary exchange format
RTOA	Convert an RZ file into FZ alpha exchange format
RTOX	Convert an RZ file into FZ binary exchange format

- Poisson suite of programs. Can be built individually or collectively using **make poisson**. Used to solve Poisson's or Laplace's equation in 2 dimensional regions (magnetostatic, electrostatic or static temperature problems).

`FORCCR`	A solver to calculate the forces.
`LATTCR`	Lattice defintion program.
`POISCR`	Solver for Poisson's or Laplace's equation.
`TRIPCR`	Postprocessor to generate a GKS metafile.

- Zebra file transfer program and associated server. See also the CSPACK manual [7] for information on configuring the servers.

`ZFTP`	The client program.
`ZSERV`	The server program.

- **VAXTAP** commands. Cannot be built using **make**. See the **VAXTAP** manual [16] for installation details.

`EINIT`	Initialise a tape with a VOL1 label written in EBCDIC.
`LABELDUMP`	Dump the VOL1 label of a tape.
`SETUP`	Mount a tape, optional STK and / or TMS support.
`STAGE`	Stage command.
`STAGECLN`	Stage space manager.
`TAPECOPY`	Copy a tape.
`WRTAPE`	Write a disk file to tape with ASCII or EBCDIC labels.
`XTAPE`	Examine the contents of a tape.

- Monitoring utilities.

`UMCOM`
`UMLOG`
`UMON`

17.8 TCP/IP considerations

Some of the CERNLIB modules require TCP/IP socket libraries. The list of these modules is defined by the symbol **need_tcp** in **makepack.com**. At the time of writing, this is defined as shown below.

List of modules requiring TCP/IP

```
$ need_tcp = ".ZFTP.ZSERV.PAWM.PAWPP.PAWSERV.TELNETG.SYSREQ.FATMEN."
```

The CERNLIB installation procedures attempt to chose the correct version of TCP/IP and act accordingly. This is done in the command file **f$tcpip.com** as follows:

Determining the TCP/IP version

```
$ If F$TRNLNM("TWG$ETC").nes.""     Then tcpip_var="Wollongong WINTCP"
$ If F$TRNLNM("MULTINET").nes.""    Then tcpip_var="MultiNet TCPIP"
$ If F$TRNLNM("UCX$NETWORK").nes."" Then tcpip_var="UCX TCPIP"
```

If you do not have one of these systems installed, then you will need to modify **f$tcpip.com** and **makepack.com** accordingly.

There are 3 areas that might require modification:

1. The selection of the appropriate include files.
2. Definitions for the C preprocessor.
3. The appropriate options file for linking.

Selecting the appropriate include files

```
$ If pack.eqs."TELNETG"
$ Then If tcppg.eqs."_W"
$      Then TCPDIR="TWG$TCP:[netdist.include"
$           assign/user_mode 'TCPDIR'],'TCPDIR'.sys],'TCPDIR'.net],-
                             'TCPDIR'.netinet],sys$library vaxc$include
$           assign/user_mode 'TCPDIR'.net]     net
$           assign/user_mode 'TCPDIR'.arpa]    arpa
$      Endif
$      If tcppg.eqs."_M"
$      Then TCPDIR="MULTINET_ROOT:[Multinet.include"
$           assign/user_mode 'RDIR'.src.cfs.cspack] arpa
$      Endif
$      If tcppg.eqs."_U"
$      Then TCPDIR="UCX??"
$      Endif
$           assign/user_mode 'TCPDIR'.sys]     sys
$           assign/user_mode 'TCPDIR'.netinet] netinet
$ Endif
```

Definitions for the C preprocessor

```
$      If tcppg.eqs."_W"   Then cco=cco+"/DEF=TWG"
$      If tcppg.eqs."_M"   Then cco=cco+"/DEF=TGV"
```

Link options file for Multinet, UCX and Wollongong

```
::::::::::::::
vmslib_m.opt
::::::::::::::
SYS$LIBRARY:vaxcrtl/share
multinet_socket_library/share


::::::::::::::
vmslib_u.opt
::::::::::::::
SYS$LIBRARY:ucx$ipc/lib
SYS$LIBRARY:vaxcrtl/share

::::::::::::::
vmslib_w.opt
::::::::::::::
CERN_ROOT:[lib]VMSLIB/lib
SYS$LIBRARY:vaxcrtl/share
```

Chapter 18: Installing CERNLIB software on VM systems

This chapter describes how CERNLIB software is installed on systems running VM/CMS. It assumes that the environment has been set up as described in section 7.3.1.

The actual software installation is performed in batch and is controlled by a service machine. Commands are sent to this service machine using the **SLIB** exec. The commands used are similar to those used on VMS systems, except that they are prefixed by **SLIB**.

18.1 Components of the CERN libraries on VM systems

The following diagram shows the structure and components of the CERN libraries on VM systems. This structure is reflected in the **MAKELIB NAMES** file.

CERNLIB structure on VM

```
//CERN/CNDIV/CERNLIB
                    /ALL
                        /CERNLIBS
                                 /KERNLIB
                                         /KERNASW
                                         /KERNNUM
                                         /KERNGEN
                                 /PACKLIB
                                         /EPIO
                                         /FFREAD
                                         /IOPACK
                                         /KAPACK
                                         /KUIP
                                         /HBOOK
                                         /MINUIT
                                         /VMIO
                                         /ZBOOK
                                         /ZEBRA
                                         /CDLIB
                                         /FATLIB
                                 /MATHLIB
                                         /LAPACK
                                         /MPA
                                         /BVSL
                                         /GEN
                                 /GRAFLIBS
                                          /GRAFLIB
                                          /GRAFGKS
                                          /GRAFGDDM
                                          /GRAFX11
                                 /PAWLIB
                                        /PAW
                                        /COMIS
                                        /SIGMA
                        /MCLIBS
                               /COJETS
                                      /COJETS
                                      /COJETSD
                               /EURODEC
                                       /EURODEC
```

```
                /EURODECD
         /HERWIG
         /ISAJET
         /JETSET
               /JETSET
               /PYTHIA
         /LEPTO
         /PHOTOS
         /PDFLIB
         /TWISTER
         /FRITIOF
         /ARIADNE
         /MCDOC
              /COJETSD
              /EURODECD
              /HERWIGD
              /ISAJETD
              /JETSETD
              /PYTHIAD
              /PHOTOSD
              /PDFLIBD
              /FRITIOFD
              /ARIADNED
/CERNPGM
         /PAWM
              /PAWGKS
              /PAWGDDM
              /PAWX11
         /KUIPC
         /FATSET
                /FATMEN
                /FATNEW
                /FATSRV
                /FATSEND
         /RZCONV
                /RTOA
                /RFRA
                /RTOX
                /RFRX
         /FLOP
         /TREE
         /ZSERV
         /PAWSERV
         /ZFTP
         /TELNETG
         /DZEDIT
                /DZEGKS
                /DZEGDDM
                /DZEX11
         /HIGZCONV
         /GRTREE
                /GRTGKS
                /GRTPS
/USERPGM
         /BANNER
         /GARFIELD
         /MAGNET
```

```
                    /POISSON
                    /TRSPRT
                    /TURTLE
```

18.2 Building standalone libraries

Complete libraries are built using the syntax **make *target***. For example, **KERNLIB** is built by typing **SLIB MAKE KERNLIB**. Again, this is similar to the VMS case, except that the **SLIB** exec sends the command to the **LIBSERV** service machine. This service machine runs the appropriate **MAKELIB** job in batch. **MAKELIB** is controlled by a configuration file **MAKELIB NAMES** on the **PUBCR 197**.

To explain how this works, let us examine the case of **KERNLIB** in more detail.

The command **MAKE KERNLIB** corresponds to an entry in the **MAKELIB NAMES** file as shown below.

MAKE_KERNLIB entry in MAKELIB NAMES file

```
***********************************************************************
* KERNLIB Library                                                     *
***********************************************************************
:NICK.MAKE_KERNLIB
:JOB.KERNLIB   :BOPT.TIME  0:10 JOBID KERNLIB
               :QSUB.
:TXTLIB.KERNASW KERNNUM KERNGEN
:FROM.  KERNASW KERNNUM KERNGEN
```

In this case, **make kernlib** will rebuild KERNLIB from the individual TXTLIBs KERNASW, KERNNUM and KERNGEN. To rebuild the individual TXTLIBs, **make from kernlib** should be used. In this case, one must ensure that the jobs terminate correctly before rebuilding KERNLIB from its components.

PACKLIB is built in a similar manner. The entry in MAKELIB names is as follows:

MAKE_PACKLIB entry in MAKELIB NAMES file

```
***********************************************************************
* PACKLIB Library                                                     *
***********************************************************************
:NICK.MAKE_PACKLIB
:JOB.PACKLIB   :BOPT.TIME  0:15 JOBID PACKLIB
               :QSUB.
:TXTLIB.CSPACK EPIO FFREAD IOPACK KAPACK KUIP HBOOK MINUIT VMIO
        ZBOOK ZEBRA KERNASW KERNNUM KERNGEN CDLIB FATLIB
:FROM.  CSPACK EPIO FFREAD IOPACK KAPACK KUIP HBOOK MINUIT VMIO
        ZBOOK ZEBRA CDLIB FATLIB
```

18.3 Building the CERNLIB TXTLIBs

- Basic libraries

`KERNLIB`	Use **slib make from kernlib** to build the components and then **slib make kernlib** to assemble the library.
`MATHLIB`	Use **slib make from mathlib** to build the components and then **slib make mathlib** to assemble the library.
`PACKLIB`	Use **slib make from packlib** to build the components and then **slib make packlib** to assemble the library.
`PAWLIB`	Use **slib make from pawlib** to build the components and then **slib make pawlib** to assemble the library.
`VKERNLIB`	As **KERNLIB**, but with vectorised code where available. Use **slib make from vkernlib** to build the components and then **slib make vkernlib** to assemble the library.
`VMATHLIB`	As **MATHLIB**, but with vectorised code where available. Use **slib make from vmathlib** to build the components and then **slib make vmathlib** to assemble the library.
`VPACKLIB`	As **KERNLIB**, but with vectorised code where available. Use **slib make from vpacklib** to build the components and then **slib make vpacklib** to assemble the library.

- Graphics libraries. Use **slib make from graflibs** and then **slib make graflibs** to build all or **slib make** *target* to build individual components.

`GRAFGDDM`	GDDM interface.
`GRAFGKS`	GKS interface.
`GRAFLIB`	Graphics kernel.
`GRAFX11`	X11 interface.

- GEANT

`GEANT314`	Geant version 3.14
`GEANT315`	Geant version 3.15
`GEANT316`	Geant version 3.16
`GEANT321`	Geant version 3.21

- Monte Carlo generators and other libraries, built using **slib make** *target*. Can be built collectively using **slib make mclibs**.

```
ARIADNE
COJETS
EURODEC
FRITIOF
HERWIG56
ISAJET65
ISAJET72
JETSET63
JETSET73
LEPTO61
PDFLIB
```

PHOTOS
TWISTER

- Sundry

CKERNEL	Emulation of C system routines.
PHTOOLS	Collection of physics tools, e.g. **FOWL**, **GENBOD** etc.

18.4 Building the CERNLIB modules

- Sundry packages

CMZ	
HTOLIB	
FCASPLIT	
BANNER	Write text in large letters.
MAKEDECK	
FLOP	Fortran coding convention checker.
HIGZCONV	Convert HIGZ RZ metafiles into postscript or GKS format.
TREE	Show calling tree of a Fortran program.
GRTGKS	Obsolete? (Convert tree to GKS metafile?)
GRTPS	Obsolete? (Convert tree to postscript file?)

- Zebra bank documentation programs.

DZEX11	Interactive program linked with X11.
DZEGKS	As above, linked with GTS-GRAL GKS.
DZEGDDM	As above, linked with IBM's GDDM software.

- FATMEN programs. Can be built individually or collectively using **slib make fatset**. See also the FATMEN manual [9] for information on configuring the FATMEN servers.

FATMEN	The FATMEN shell.
FATNEW	Program to build a new FATMEN catalogue.
FATSEND	Program to transfer FATMEN updates between servers.
FATSRV	The FATMEN server.

- GARFIELD programs.

GARFIELD
GARFRUN

- HEPDB programs. Can be built individually or collectively using **make hepdbset**. See also the HEPDB manual [11] for information on configuring the HEPDB servers.

HEPDB	The interactive interface.
CDSERV	The HEPDB server.
CDMAKE	Program to make a new database.
CDMOVE	Program to move updates between server queues.

- KUIP programs.

KUIPC	The KUIP compiler.

- The PAW clients and server. Can be built individually or collectively, with the exception of **PAWSERV**, using **make pawm**.

PAWX11	PAW linked with the X11 libraries and the TCP/IP libraries.
PAWGDDM	PAW linked with IBM's GDDM graphics package and the TCP/IP libraries.
PAWGKS	PAW linked with GTS-GRAL GKS and the TCP/IP libraries.
PAWSERV	The PAW server

- Programs to permit transfer of RZ files. These may be built individually, e.g. **make rfra**, or collectively using **make rzconv**.

RFRA	Convert an RZ file from FZ alpha exchange format
RFRX	Convert an RZ file from FZ binary exchange format
RTOA	Convert an RZ file into FZ alpha exchange format
RTOX	Convert an RZ file into FZ binary exchange format

- Poisson suite of programs. Can be built individually or collectively using **make poisson**. Used to solve Poisson's or Laplace's equation in 2 dimensional regions (magnetostatic, electrostatic or static temperature problems). These programs are on the **MAGNET** disk on CERNVM (**GIME MAGNET**).

FORCCR	A solver to calculate the forces.
LATTCR	Lattice defintion program.
POISCR	Solver for Poisson's or Laplace's equation.
TRIPCR	Postprocessor to generate a GKS metafile.

- Zebra file transfer program and associated server. See also the CSPACK manual [7] for information on configuring the servers.

BZFTP	The client program, *batch* version. Program will exit with a unique return code in case of problems..
ZFTP	The client program.
ZSERV	The server program.

Chapter 20: Installing CERNLIB software on Windows/NT systems

Chapter 22: Installing CERNLIB on a Unix system that is not already supported

If you try to install the CERN Program Library on a system that is not currently supported, you will get an error from makepack as shown below. Should this occur, follow the procedure below. Please remember to provide your feedback to the CERN Program Library office so that the work is not lost and can be shared by others.

Typical errors from makepack on unsupported Unix systems

```
zfatal:/cernlib/cern/94a/mgr (59) make -n kernlib

...

Make: make: 1254-002 Cannot find a rule to create target
     /cernlib/cern/94a/src/car/kernxxx.car from dependencies.   <---
   Stop.
```

- Modify the script **cernsys** to set the variables **PLIUWC**, **PLINAME** and **PLISYS** as appropriate. The names chosen for **PLINAME** and **PLIUWC** should be agreed with the CERN Program Library office, if possible.

  `PLIUWC` Three letter code indicating the machine type. This code is used to find the system dependant part of **KERNLIB**. e.g., for the RS6000, **PLIUWC** is set to **irs** and the RS6000 specific part of **KERNLIB** is in **kernirs.car**.

  `PLINAME` The appropriate **PATCHY** flag. This will be automatically selected by the CERNLIB installation jobs.
  If we were installing the libraries under a flavour of Unix known as **OBELIX**, we would add the following line to the **cradle**.

  Selecting Unix code on the OBELIX system

  ```
  +USE,UNIX,IF=OBELIX.
  ```

  `PLISYS` This should be set to **SYSTEMV**, **BSD**, **MACH** or **UNKNOWN** as appropriate.

- Chose a machine specific KERNLIB pam for a similar system and copy it, e.g. to **kernobx.car** in the case of **OBELIX**.

Chapter 23: Installing CERNLIB software on other systems

Should you wish to install the CERN program library on a machine to which it has not already been ported, the following tips may prove useful.

23.1 Starting point

Start from a system as close as possible to the new system. For example, if you were porting the library to Alpha/VMS, an appropriate starting point would be the VAX/VMS version.

23.2 File naming conventions

Most Unix systems use **.f** for Fortran files, although the Apollo uses **.ftn**.

23.3 Compiler name and options

The Fortran compiler is typically invoked using the **f77** command on Unix systems, although the RS6000 uses **xlf** and the Convex **fc**.

23.4 Porting PATCHY

As the CERNLIB installation procedures currently use **PATCHY**, you will either have to port **PATCHY** and possibly also the splitting program **FCASPLIT**, or extract the code on a system to which these programs have already been ported.[1]

If it is necessary to modify the compiler and/or options, one should also remove the check of the file **p4boot.sh** against **p4boot.sh0**. If there is a mismatch, the installation procedure will exit.

Fortran installation packages. It may be necessary to make modifications to the files **rceta.f** or **fcasplit.f**

23.5 Likely areas of incompatibility

There are a number of areas where incompatibilities between machines are likely to arise. These include:

1. Fortran **OPEN** statements. Modifications are likely to routines such as **KUOPEN**, **RZOPEN**, **FMOPEN** etc. In addition to various language extensions, such as the **READONLY** and **SHARED** attributes in VAX Fortran, the units in which the record length of direct access files often varies (typically bytes or words).

2. The syntax for file and directory names is likely to differ. This will affect packages such as CSPACK, FATMEN and HEPDB amongst others.

3. Data representation. The majority of new systems support IEEE floating point. If your system does not support IEEE floating point format, then you will need to modify the KERNLIB package **IE3CONV**. If your system uses a floating point format that already exists, then you should find the appropriate code in one of the KERNLIB pam files. For example, the routines to convert to and from IBM floating point representation can be found in the **KERNIBM** pam file.

[1] The latter technique was used to install the CERN libraries on Windows/NT. **PATCHY** was run on a PC running MSDOS and the output files written to a Novell server, from where they could be accessed from a Windows/NT system.

4. Byte order. Most systems are *big endian*, which corresponds to the way that we write numbers in every day life (i.e. the left most bit has the highest significance. Some systems, in particular DEC systems (VAX, Alpha, Ultrix) and IBM PCs and compatibles, are *little endian*.

5. Interface to the system. Routines in the KERNLIB package **CINTF** will probably require modification.

6. The graphics packages may require heavy modification depending on the graphics facilities on the target machine.

23.6 Porting the CERN libraries from Sun OS to Solaris

The following modifications were required to port the CERN libraries from Sun OS to Solaris.

Cradle for KERNGEN

```
+EXE.
+USE,*KERNGEN,$PLINAME.
+ASM,22    ,IF=CRAY,IBMVM,VAXVMS.
+ASM,23,T=A,IF=IBMVM.kerngensh.sh
+ASM,24.
+ASM,31,T=A.:kerngen2F.f
+USE,QSYSBSD ,T=I,IF=SOLARIS.
+USE,QENVBSD ,T=I,IF=SOLARIS.
+USE,QSIGJMP   ,IF=SOLARIS.
+USE,QGETCWD   ,IF=SOLARIS.
+USE,QSIGPOSIX ,IF=SOLARIS.
+DIV,P=TCGEN,D=UCOPY2,IF=SOLARIS.
+DEL,P=SUNGS,D=JUMPAD,C=1-9,IF=SOLARIS.
+DEL,P=SUNGS,D=JUMPX2,C=1-46,IF=SOLARIS.
+PAM,11,T=C,A.$CERN_ROOT/src/car/kerngen
+PAM,12,T=C,A.$CERN_ROOT/src/car/kern$PLIUWC
+PAM,13,T=C,A,IF=IBMVM.$CERN_ROOT/src/car/kerncms
+PAM,14,T=C,A.$CERN_ROOT/src/car/kernfor
+QUIT.
```

The above cradle has been slightly simplified for clarity. However, we see that the main changes have been the selection of certain flags that characterise the operating system. These flags are described in appendix B on page 107.

We repeat those selected for Solaris below.

`QSYSBSD`	Unix system BSD (system 5 otherwise)
`QSIGJMP`	Posix sigsetjmp/siglongjmp for setjmp/longjmp
`QENVBSD`	BSD setenv is available
`QSIGBSD`	signal handling with BSD sigvec
`QSIGPOSIX`	signal handling with Posix sigaction

23.7 Porting CERNLIB to FACOM VPX series

The FACOM VPX series run a Unix System V system. However, the floating point representation is that of IBM mainframes.

We start with the Sun Solaris versions of the libraries, e.g. **KERNSUN** with the flag **SOLARIS**.

In ZEBRA, we must ensure that data is correctly converted on input and output.

For IBM mainframes, the required definitions are in the deck **IBM** of patch **FQ** in the Zebra pam file. The conversion of data on input and output is performed in the routines **FZICV** and **FZOCV** respectively. The conversion is performed by sequences as shown below (plus the corresponding sequences for input). For the FACOM, the following is probably sufficient:

Selecting correction input/output conversion

```
+USE, FQIE3FSC.    use default CALL IE3FOS for output single prec.
+USE, FQIE3FDC.    use default CALL IE3FOD for output double prec.
+USE, FQIE3TSC.    use default CALL IE3FOS for input single prec.
+USE, FQIE3TDC.    use default CALL IE3FOD for input double prec.
```

which will call the **KERNLIB** conversion routines (which must of course be provide) for floating point data and copy as is for all other data. (The VPX series uses the ASCII character set and is big endian).

In this respect, the PATCH **IBX**, which is for AIX on IBM mainframes, and **KERNIBX CAR**, which contains Fortran versions of the floating point conversion routines, may work directly on the FACOM.

`FZOCVFB`	Output conversion of bit strings
`FZOCVFI`	Output conversion of integer data
`FZOCVFF`	Output conversion of single precision data
`FZOCVFD`	Output conversion of double precision data
`FZOCVFH`	Output conversion of hollerith data

Chapter 24: Rebuilding components of the library *by hand*

On occasion, one may need to rebuild a component of the CERN library. This typically happens when there is a different version of the operating system, compiler or shared library on the local system to that used at CERN. Alternatively, one may wish to link to a licensed product that is either not available at CERN or cannot be distributed. A typical example is PAW.

24.1 Rebuilding PAW on VMS systems

Example of rebuilding PAW on VMS systems

```
$!
$! Build various versions of PAW
$!
$ set noon
$!
$ architecture=f$edit(f$getsyi("ARCH_NAME"),"UPCASE")
$!
$ if architecture .eqs. "ALPHA"
$    then
$       cc:==cc/standard=vaxc
$       link:==link/nonative
$    endif
$!
$  ypatchy cern:['cern_level'.src.car]paw.car pawmain.for :go
+USE,CZ.
+USE,MAIN.
+USE,P=PAW,D=0PAMAIN.
+EXE.
+KEEP,PAWSIZ.
      PARAMETER (NWPAW=500000)
+PAM,T=C.
+QUIT.
$!
$  ypatchy cern:['cern_level'.src.car]paw.car pawpp.for :go
+USE,CZ.
+USE,MAIN,MOTIF.
+USE,P=PAW,D=0PAMAINM.
+EXE.
+KEEP,PAWSIZ.
      PARAMETER (NWPAW=500000)
+PAM,T=C.
+QUIT.
$!
$  create czdummy.for
      subroutine czdummy
      entry czopen
      entry czclos
      entry czputa
      entry czgeta
      entry czputc
      entry czgetc
      entry cztcp
      entry CONNECT
      entry GETHOSTBYNAME
```

```
      entry GETSERVBYNAME
      entry HTONS
      entry INET_ADDR
      entry RECV
      entry SELECT
      entry SEND
      entry SETSOCKOPT
      entry SHUTDOWN
      entry SOCKET
      entry multinet_get_socket_errno_addr
      entry socket_close
      entry socket_ioctl
      entry socket_perror
      end
$!
$  create gethostname.c
/*
 * return the node name
 */
#include <descrip.h>
#include <lnmdef.h>
int
gethostname( node, len )
char *node;
int len;

  $DESCRIPTOR( tabnam, "LNM$SYSTEM" );
  $DESCRIPTOR( lognam, "SYS$NODE" );
  int length = 0;
  struct
    short buffer_length;
    short item_code;
    char  *buffer_address;
    int    *return_length;
    int    item_list_end;
   itmlst;     /* Disabled auto initialization =  len - 1,
   LNM$_STRING, node, &length, 0 ; */
  char *p = node;

    /* Manual initialization code inserted by CRL on 931206 */
  itmlst.buffer_length   = (len - 1);
  itmlst.item_code       = LNM$_STRING;
  itmlst.buffer_address  = node;
  itmlst.return_length   = &length;
  itmlst.item_list_end   = 0;

  sys$trnlnm( 0, &tabnam, &lognam, 0, &itmlst );

  while( p[0] != '
0' && p[0] != ':' )
    p++;
  p[0] = '
0';

  return( 0 );

$!
```

```
$! Compile the main program(s) if found
$!
$  if f$search("PAWMAIN.FOR") .nes. ""
$     then
$     write sys$output "Compiling PAWMAIN..."
$     fortran pawmain
$  endif
$!
$  if f$search("PAWPP.FOR") .nes. ""
$     then
$     write sys$output "Compiling PAWPP..."
$     fortran pawpp
$  endif
$!
$  if f$search("GETHOSTNAME.C") .nes. ""
$     then
$     write sys$output "Compiling GETHOSTNAME..."
$     cc gethostname
$  endif
$!
$  if f$search("CZDUMMY.FOR") .nes. ""
$     then
$     write sys$output "Compiling CZDUMMY (dummy TCP/IP routines)..."
$     fortran czdummy
$  endif
$!
$! Linking of PAW/GKS version
$!
$! Licensed software required: GTS-GRAL GKS (installed and distributed by CERN)
$!
$   if architecture .eqs. "ALPHA"
$      then
$         write sys$output "PAW/GKS only available on VAX/VMS"
$      else
$         write sys$output "Link GTS-GRAL GKS version of PAW..."
$         cernlib pawlib,mathlib,packlib,graflib,packlib
$         link/exe=pawgks pawmain,czdummy,'LIB$
$      endif
$!
$! Linking of PAW/DEC-GKS
$! Licensed software required: DEC-GKS from Digital
$!
$   write sys$output "Link DEC-GKS version of PAW..."
$   cernlib pawlib,mathlib,packlib,graflib/dgks,packlib
$   link/exe=pawdgks pawmain,czdummy,'LIB$
$!
$! Linking of PAW/X11
$! Licensed software required: Motif 1.1
$!
$   write sys$output "Link PAW/X11..."
$   cernlib pawlib,mathlib,packlib,graflib/x11,packlib
$   link/exe=pawx11 pawmain,gethostname,czdummy,'LIB$
$!
$! Linking of PAW/X11_M
$! Licensed software required: Motif 1.1, Multinet TCP/IP
$!
$   write sys$output "Link PAW/X11_M..."
```

```
$   cernlib pawlib,mathlib,packlib,graflib/x11,packlib
$   if architecture .eqs. "ALPHA"
$      then
$         link/exe=pawx11_m -
            pawmain,'LIB$',sys$input/opt
            multinet_socket_library/share
            sys$library:decc$shr/share
$      else
$         link/exe=pawx11_m -
            pawmain,'LIB$',sys$input/opt
            multinet_socket_library/share
            sys$library:vaxcrtl/share
$   endif
$!
$! Linking of PAW/X11_U
$! Licensed software required: Motif 1.1, DEC TCP/IP (UCX)
$!
$   write sys$output "Link PAW/X11_U..."
$   cernlib pawlib,mathlib,packlib,graflib/x11,packlib
$   if architecture .eqs. "ALPHA"
$      then
$         link/exe=pawx11_u -
            pawmain,'LIB$',sys$library:ucx$ipc/lib,sys$input/opt
            sys$library:ucx$ipc_shr/share
            sys$library:decc$shr/share
$      else
$         link/exe=pawx11_u -
            pawmain,'LIB$',sys$library:ucx$ipc/lib,sys$input/opt
            sys$library:ucx$ipc_shr/share
            sys$library:vaxcrtl/share
$   endif
$!
$! Linking of PAW/X11_DECW
$! Licensed software required: DECWindows
$!
$   write sys$output "Link PAW/X11_DECW..."
$   if architecture .eqs. "ALPHA"
$      then
$         write sys$output "This option only available on VAX/VMS"
$      else
$         lib$:==CERN:['cern_level'.LIB]PAWLIB/LIB,PACKLIB/LIB,MATHLIB/LIB,-
GRAFLIB/LIB,GRAFX11/LIB,PACKLIB/LIB,KERNLIB/LIB"
$         link/exe=pawx11_decw -
            pawmain,gethostname,czdummy,'LIB$',sys$input/opt
            cern:[decw]decw$xlibshr/share
            cern:[decw]decw$dwtlibshr/share
            cern:[decw]decw$transport_common/share
            sys$library:vaxcrtl/share
$   endif
$!
$! Linking of PAW/X11_DECW_M
$! Licensed software required: DECWindows, Multinet
$!
$   write sys$output "Link PAW/X11_DECW_M..."
$   if architecture .eqs. "ALPHA"
$      then
$         write sys$output "This option only available on VAX/VMS"
```

```
$      else
$         lib$:==CERN:['cern_level'.LIB]PAWLIB/LIB,PACKLIB/LIB,MATHLIB/LIB,-
GRAFLIB/LIB,GRAFX11/LIB,PACKLIB/LIB,KERNLIB/LIB"
$         link/exe=pawx11_decw_m -
            pawmain,'LIB$',sys$input/opt
            multinet_socket_library/share
            cern:[decw]decw$xlibshr/share
            cern:[decw]decw$dwtlibshr/share
            cern:[decw]decw$transport_common/share
            sys$library:vaxcrtl/share
$   endif
$!
$! Linking of PAW++ with Multinet
$!
$   write sys$output "Link PAW++ with Multinet..."
$   cernlib pawlib,mathlib,packlib,graflib/motif,packlib
$   if architecture .eqs. "ALPHA"
$      then
$         link/exe=pawpp -
            pawpp,'LIB$',sys$input/opt
            multinet_socket_library/share
            sys$library:decc$shr/share
$      else
$         link/exe=pawpp -
            pawpp,'LIB$',sys$input/opt
            multinet_socket_library/share
            sys$library:vaxcrtl/share
$   endif
$!
$! Linking of PAW++ with DEC TCP/IP (UCX)
$!
$   write sys$output "Link PAW++ with UCX..."
$   cernlib pawlib,mathlib,packlib,graflib/motif,packlib
$   if architecture .eqs. "ALPHA"
$      then
$         link/exe=pawpp_u -
            pawpp,'LIB$',sys$input/opt
            sys$library:ucx$ipc_shr/share
            sys$library:decc$shr/share
$      else
$         link/exe=pawpp_u -
            pawpp,'LIB$',sys$library:ucx$ipc/lib,sys$input/opt
            sys$library:ucx$ipc_shr/share
            sys$library:vaxcrtl/share
$   endif
$!
$!
$! Linking of PAW++ without Multinet
$!
$   write sys$output "Link PAW++ without Multinet..."
$   cernlib pawlib,mathlib,packlib,graflib/motif,packlib
$   if architecture .eqs. "ALPHA"
$      then
$         link/exe=pawpp_dnet -
            pawpp,czdummy,gethostname,'LIB$',sys$input/opt
            sys$library:decc$shr/share
$      else
```

```
$         link/exe=pawpp_dnet -
             pawpp,czdummy,gethostname,'LIB$',sys$input/opt
             sys$library:vaxcrtl/share
$   endif
```

24.2 Relinking PAW on VM/CMS systems

The following section describes how to relink PAW on systems running VM/XA or VM/ESA. It assumes that the CERN libraries, e.g. PACKLIB, PAWLIB, are already installed on your system.

24.2.1 General requirements

- VS-FORTRAN 2.x (our libraries are generated with 2.5.0)
- IBM C/370 2.1.0 (not compatible with Waterloo C)
- TCP/IP 2.1 (if you want to use the "rlogin" facility in PAW)
- GKS, GDDM or X11 graphics libraries
- C run time library

24.2.2 Extracting the main program

The main program can be extracted as shown below:

Extracting the PAW main program

```
/* Extract PAWMAIN */
/* PAWMAIN EXEC    */
queue "+USE,CZ."
queue "+USE,MAIN."
queue "+USE,P=PAW,D=0PAMAIN."
queue "+EXE."
queue "+KEEP,PAWSIZ."
queue "      PARAMETER (NWPAW=500000)"
queue "+PAM,11,T=C."
queue "+QUIT."
exec        ypatchy,
            'pam="PAW CAR *"',
            'asm="PAWMAIN FORTRAN"'
```

This results in the following Fortran file:

PAW main program

```
*CMZ :  2.04/08 30/11/93  14.07.07  by  Rene Brun
*-- Author :    Rene Brun   03/01/89
      PROGRAM PAMAIN
*
*         MAIN Program for basic PAW
*
      PARAMETER (NWPAW=500000)
*
      COMMON/PAWC/PAWCOM(NWPAW)
*
      CALL PAW(NWPAW,IWTYP)
*
      CALL KUWHAG
*
      CALL PAEXIT
*
      STOP
      END
      SUBROUTINE QNEXT
      END
```

The ENDMODU routine shown below is used in order to reduce the size of the PAW module.

ENDMODU FORTRAN

```
      BLOCK DATA ENDMODU
      END
```

24.2.3 Building the GKS version of PAW

This requires the GTS-GRAL GKS software, which is installed and distributed by CERN.

Building the GKS version of PAW

```
 VFORT PAWMAIN
 VFORT ENDMODU
 CERNLIB PAWLIB GRAFLIB ( GTS2D LINK
 LOAD PAWMAIN ( CLEAR NOAUTO
 INCLUDE ENDMODU
 GENMOD PAWGKS  ( FROM PAMAIN TO ENDMODU RMODE ANY AMODE ANY
```

24.3 Building the GDDM version of PAW

This requires the GDDM software from IBM.

Building the GDDM version of PAW

```
VFORT PAWMAIN
VFORT ENDMODU
CERNLIB PAWLIB GRAFLIB ( GDDM LINK
LOAD PAWMAIN ( CLEAR NOAUTO
INCLUDE ENDMODU
GENMOD PAWGDDM ( FROM PAMAIN TO ENDMODU RMODE ANY AMODE ANY
```

24.4 Building the X11 version of PAW

This requires IBM's X11 software, which is bundled together with TCP/IP.

Building the X11 version of PAW

```
VFORT PAWMAIN
VFORT ENDMODU
CERNLIB PAWLIB GRAFLIB ( X11 LINK
LOAD PAWMAIN ( CLEAR NOAUTO
INCLUDE ENDMODU
GENMOD PAWX11  ( FROM PAMAIN TO ENDMODU RMODE ANY AMODE ANY
```

Part IV

CERNLIB – Network Installation Procedures

Chapter 25: Network installation procedures

Various procedures exist to copy all or part of the CERN Program Library over the network. Below we provide pointers to some of these procedures.

25.1 CERN_MANAGE

CERN_MANAGE is a package written by Mike Kelsey. The www html entry is listed below.

CERN_MANAGE html file

```
CERN_MANAGE

Freehep Name
   CERN_MANAGE
Version
   1.06
Date
   Jan.  17, 1994
Title
   Management of Remote Site CERN Software
Author(s)
   KELSEY Michael H.
Contact
   KELSEY, Michael H.  (kelsey@cithex.caltech.edu)
Subject Areas
   networking_email_news , general_libraries
News Group or Email
   cern.heplib, kelsey@cithex.caltech.edu.  There is also a
   mailing list.  To subscribe, send mail to
   listserv@cithex.caltech.edu, with the line 'SUBSCRIBE
   CERN_MANAGE' in the body of the message (or 'HELP' for details
   on this mail robot)
Bug Reports to
   kelsey@cithex.caltech.edu
Software Needed
   VAX/VMS 5.5-2 or later, or Alpha/VMS.
Hardware Needed
   VAX workstation, minicomputer or mainframe DECNET access to
   VXCERN::
Access
   Anonymous FTP from:
   ftp://cithe501.cithep.caltech.edu/pub/cern_manage
   ftp://freehep.scri.fsu.edu/freehep/networking_email_news/cern_manage
   DECnet from: CITHEX::CERN:[CERN_MANAGE]
User Base
   Managers/installers of CERN software on non-CERN VAXen: Brown
   University, Caltech, Drexel, GANIL (France), INFN Bologna,
   Mainz (Germany), Michigan, SLAC, TRIUMF, UCLA, Wisconsin, and
   more...
Documentation
   README (ASCII, introduction and purpose) cern_manage.txt
   (ASCII) cern_manage.html (Hypertext/WWW format, see link below)
Published References
   NONE
Freehep Directory
```

```
   networking_email_news/cern_manage
More Information

See also CERNLIB .

Abstract

CERN_MANAGE is a package of VMS command procedures useful for
maintaining the CERN software environment on a non-CERN VAX or
VAXcluster.  It was developed under VMS 5.5-2, and uses no
third-party or binary-executable code, other than native VAX/VMS
commands.

The CERN_MANAGE system will make suitable additions to the VMS
system environment to support the full range(*) of CERN-defined
commands, as documented by CERN.

(*) Tests have been made using the commands CERNLIB, PAW, RFRA,
RTOA, RFRX, RTOX, and the YPATCHY series.  Other CERN-defined
commands have not been tested due to lack of necessary software or
equipment.

Latest Modifications:

New routine, SYNCH_RDT.COM, used to override the default local
file modification date with the date of the file at CERN, to avoid
time-zone problems in incremental updates.
```

Part V

Appendices

Appendix A: Setting the PLINAME variable

The CERNLIB cradles contain a **USE** statement for **$PLINAME**. This is replaced by the value of the environmental variable **PLINAME** by the utility **YEXPAND**.

The following list describes the meanings of the various values that can be assigned to **PLINAME**.

PLINAME may be set to more than one value. For example, on VXCERN **PLINAME** is set as follows:

Example of setting PLINAME

```
$ dnetarea=f$getsyi("node_area")
$ _la=(dnetarea.eq.22.or.dnetarea.eq.23)
$ _axp=f$getsyi("arch_name").eqs."Alpha"
$ if f$type(pliname) .eqs. ""
$ then
$   if _la
$     then
$       setenv PLINAME "VAX,VAXVMS,CERN"      ! at CERN
$     else
$       setenv PLINAME "VAX,VAXVMS"           ! elsewhere
$     endif
$   if _axp then setenv PLINAME 'PLINAME',QMALPH
$ endif
```

Thus, PLINAME will be set to "VAX,VAXVMS,CERN" for systems in DECnet areas 22 or 23 and "VAX,VAXVMS" elsewhere. If the system is an Alpha, then ",QMALPH" will be added to this string.

PLINAME should contain one or more of the following strings, as appropriate.

`IBMVM`	IBM mainframes running VM/CMS
`IBMMVS`	IBM mainframes running MVS
`CONVEX`	By itself, implies 64 bit version of the libraries for Convex. To get the 32 bit version, use also **SINGLE**.
`IBM`	IBM mainframe - selects features generic to both MVS and VM/CMS
`SLACBATCH`	Activates code specific to the SLAC Batch system for VM/CMS systems.
`ALLIANT`	Indicates Alliant computer. If used in conjunction with **QMINTEL**, implies Alliant with Intel processor.
`AMIGAUX`	Amiga Unix
`APOLLO`	Apollo workstation with the **ftn** compiler. If used in conjunction with **APOF77**, then the version appropriate for the **f77** compiler will be generated.
`CDC`	Control Data systems
`CRAY`	Cray computers
`DGE`	Data General computers
`MSDOS`	PCs running MSDOS
`DECS`	DECstations running Ultrix. If used in conjunction with **QMVAOS**, implies Alpha workstations running **OSF**
`DECOSF`	*New flag for Alphas running OSF*
`HPUX`	HP workstations running HP/UX.
`IBMAIX`	IBM mainframes running AIX.

`IBMRT`	IBM Risc processors (RT, RS) running AIX.
`CERN`	Select CERN specific features
`LINUX`	PCs running the LINUX operating system.
`MACMPW`	Macintosh computers.
`NORD500`	The NORD 500 series of computers.
`NECSX`	NEC SX computer.
`NEXT`	NeXT workstations.
`SGI`	Silicon Graphics workstations.
`SHIFT`	Activate code specific to systems running the CORE/SHIFT software.
`SUN`	Sun workstations.
`TMO`	Transputer with Meiko compiler.
`VAXVMS`	VMS systems. If used with **QMALPH**, means Alpha processors.
`WINNT`	Windows/NT systems.

Appendix B: PATCHY/CMZ flags and their meanings

The following information was extracted from the **KERNFOR** pam file.

B.1 Flags for different computer types

`QMNNB32` for an unknown 32-bit machine

`QMALPH` Alpha eXtended processor (AXP)

`QMALT` Alliant
`QMAMX` Amiga Unix

`QMAPO` Apollo
`QMAPO10` Apollo DPS 10000

`QMCDCV` CDC 6000/7000/Cyber with Fortran 5
`QMCDC` CDC 6000/7000/Cyber with Fortran 4

`QMDOS` MS-DOS with F2C + G cc compilers
`QMNDP` MS-DOS with NDP Fortran

`QMCRY` CRAY systems COS or UNICOS
`QMCRU` CRAY system UNICOS only

`QMCVX` General Convex flag
`QMCV64` Convex 64-bit mode
`QMCV32` Convex 32-bit mode

`QMDGE` Data General, ECLIPSE

`QMHPX` Hewlett Packard HP Unix

`QMIBM` IBM 360 / 370
`QMIBMVF` IBM Vector facility
`QMIBMXA` IBM Xtended Adressing
`QMIBMFSI` Fortran Siemens **OBSOLETE - use QMFIBMSI**
`QMIBMFVS` Fortran VS **OBSOLETE - use QMFIBMVS**

`QMIBX` IBM 3090 with system AIX
`QMIRT` IBM / RT and 6000 with xlf compiler

`QMND3` NORD 500
`QMNXT` Next

`QMPDP`	DEC PDP 10
`QMSGI`	Silicon Graphics
`QMSUN`	SUN
`QMTMO`	Transputer with Meiko compiler
`QMUNI`	UNIVAC 1100 with earlier compilers
`QMUNO`	UNIVAC 1100 with FTN compiler
`QMVAX`	Digital VAX
`QMVMI`	DECstation (MIPS processor unless QMALPH is also specified, then Alpha processor).
`QMVAO`	Digital Alpha with OSF, + S for 32-bit (+ L for 64-bit later)
`TYPE`	Force strong typing, i.e. **IMPLICIT NONE**

B.2 Flags to indicate word capacity

`B32`	32 bits in one computer word
`B36`	36 bits in one computer word
`B48`	48 bits in one computer word
`B60`	60 bits in one computer word
`B64`	64 bits in one computer word
`B36M`	36 bits or more per word
`B48M`	48 bits or more per word
`B60M`	60 bits or more per word
`A4`	4 characters in 1 computer word
`A5`	5 characters in 1 computer word
`A6`	6 characters in 1 computer word
`A8`	8 characters in 1 computer word
`A10`	10 characters in 1 computer word
`A5M`	5 characters or more per word
`A6M`	6 characters or more per word
`A8M`	8 characters or more per word

B.3 Flags for other computer or Fortran features

`QF2C`	Compiled using F2C and gcc
`QFDEC`	Compiled using DEC Fortran
`QFNDP`	Compiled using NDP Fortran
`QMFIBMSI`	Fortran Siemens
`QMFIBMVS`	Fortran VS

`QFMSOFT` Compiled using Microsoft Fortran

`QASCII` Character set is ASCII
`QEBCDIC` Character set is EBCDIC

`QIEEE` Floating point representation is IEEE

`QISASTD` ISA standard intrinsic functions available : IAND, IOR, NOT, ISHFT
`QMILSTD` MIL standard intrinsic functions available : IBITS, MVBITS, ISHFTC

`QHOLL` Hollerith constants exist
`EQUHOLCH` EQUIVALENCE Hollerith/Character ok
`QORTHOLL` orthodox Hollerith storage left to right in word

`QSYSBSD` Unix system BSD (system 5 otherwise)
`QSIGJMP` Posix sigsetjmp/siglongjmp for setjmp/longjmp
`QENVBSD` BSD setenv is available
`QGETCWD` BSD getwd is not available, but getcwd is available
`QSIGBSD` signal handling with BSD sigvec
`QSIGPOSIX` signal handling with Posix sigaction

`QX_SC` external names are lower case with underscore
`QXNO_SC` external names are lower case without underscore
`QXCAPT` external names are capital

`QCCINDAD` routine entry addresses are passed double indirect in Fortran calls (needed in C routines)

`INTDOUBL` use double precision for some internal calculations (used at present only in the TR routines)

`NOSHIFT` left/right shift is not available, sequence QSSHIFT cannot be defined

`HEX` dumps must be done in hexadecimal representation else: dumps are in octal

`ENTRET` multiple entry functions must return by entry name else: return by function name works ok

`ENTRCDC` CDC Fortran 4 syntax for ENTRY statement else: ENTRY statement contains argument list

B.4 Flags inherited from KERNNUM - only used in P=TCNUM

`NUMLOPRE` floating point precision for 32-bit machines

`NUMHIPRE` =-NUMLOPRE

`NUME293` maximum exponent = 10**293

`NUME75` maximum exponent = 10**75

`NUME38` maximum exponent = 10**38

Appendix C: Reserved prefixes

Most of the routines in the CERN library begin with one or two letters (three in the case of HPLOT) that identify the package. Collaborations wishing to avoid clashes may register two character prefixes with the CERN Program Library office.

N.B. the letter Z should be avoided as second character as this is used by ZEBRA routines.

The following convention is used.

A		
B		
C	CD	HEPDB routines
	CS	COMIS routines
	CZ	Client routines of CSPACK
D	DZ	ZEBRA debug and documentation routines
E	EP	EPIO routines
F	FA	FATMEN internal routines and common blocks
	FF	FFREAD routines
	FM	FATMEN user callable routines
	FM	ZEBRA sequential I/O (FZ) routines
G	G	GEANT routines
H	H	HBOOK routines
	HPL	HPLOT routines
I		
J	JZ	The ZEBRA jump package
K	K	KUIP routines
	KA	KAPACK routines
L	LZ	ZEBRA utility functions
M	MZ	ZEBRA memory management routines
N		
O		
P		
Q		
R	RZ	ZEBRA random access (RZ) I/O routines
S	SI	SIGMA
	SZ	CSPACK server routines
T	TZ	ZEBRA titles package
U		
V	VM	VMIO routines
W		
X	XZ	CSPACK remote I/O routines
Y		
Z	Z	ZBOOK routines
	LZ	ZEBRA utility routines

Appendix D: Organization of KERNLIB

KERNLIB routines in machine language or otherwise special for individual machines are found on the following PAM-files:

`KERNALI` for Alliant with Intel processor
`KERNALT` for Alliant
`KERNAMX` for Amiga/UX machines
`KERNAPO` for APOLLO
`KERNA10` for APOLLO
`KERNCDC` for CDC 7600 / 6000
`KERNCMS` for IBM systems running VM/CMS
`KERNCRY` for CRAY RESEARCH INC.
`KERNCRU` for CRAY RESEARCH INC. running Unicos (copy or link to the above)
`KERNCVX` for Convex
`KERNDGE` for DATA GENERAL
`KERNDOS` for MS-DOS systems with NDP Fortran or f2c+gcc
`KERNHPX` for HP Unix
`KERNHYW` for Honeywell/Bull
`KERNIBM` for IBM 3090 with systems MVS or VM
`KERNIBX` for IBM 3090 with system AIX
`KERNIRS` for IBM / RS6000
`KERNIRT` for IBM / RT (a copy or link to the above)
`KERNLNX` for systems running the LINUX operating system using f2c
`KERNMPW` for MAC II machines, MPW shell, LSE Fortran
`KERNNOR` for NORD 500
`KERNNXT` for NeXT
`KERNOS9` for OS9
`KERNOSF` for Alpha OSF (copy or link to **KERNVMI**)
`KERNPDP` for DEC PDP 10
`KERNSGI` for Silicon Graphics
`KERNSUN` for SUN
`KERNSOL` for Solaris (a copy or link to the above)
`KERNTMO` for Transputer with Meiko compiler
`KERNUNI` for UNIVAC 1100 SERIES
`KERNVAX` for Digital VAX 11
`KERNVMI` for Digital VAX with MIPS processor
`KERNWIN` for Windows NT (a copy or link to **KERNDOS**)

In addition to these machine specific PAM files,

`KERNBIT` PAM for CN/ASD routines
`KERNFOR` Machine independant routines
`KERNGEN` PAM containing flags patch to select correct versions of KERNLIB
`KERNNUM` Numerical section of KERNLIB

Appendix E: Examples of Transarc naming conventions

The Andrew Distributed File System, commonly known as **afs**, supports a special directory name known as **@sys**. If this is specified in a file or directory name, it will be automatically translated as shown in the table below. **N.B. this table is not a complete list of all supported system types**. The same convention is followed on the asisftp ftp server at CERN.

If you have **afs** installed on your system, you may obtain the correct value of **@sys** as shown below:

Obtaining the value of @sys

```
zfatal:/home/cp/jamie/cernlib/doc (21) fs sysname
Current sysname is 'rs_aix32'
```

Digital machines

`pmax_ulx`	DECstations with the MIPS processor running Ultrix
`vax_ul4`	VAXes running Ultrix

Hewlett-Packard machines

`hp300_ux70`	HP 9000 series 300/400 running HP/UX 7.0
`hp300_ux80`	HP 9000 series 300/400 running HP/UX 8.0
`hp700_ux805`	HP 9000 series 700 running HP/UX 8.0.5
`hp700_ux807`	HP 9000 series 700 running HP/UX 8.0.5
`hp700_ux90`	HP 9000 series 700 running HP/UX 9

IBM machines

`rs_aix31`	IBM RS/6000 running AIX 3.1
`rs_aix32`	IBM RS/6000 running AIX 3.1
`rt_aix221`	IBM RT/PC running AIX 3.1

Sun machines

`sun4c_411`	Sun SPARCStations running Sun OS 4.1.1 or 4.1.2
`sun4m_412`	Sun 600 series MP SPARCStations running Sun OS 4.1.2

Appendix F: The CERN automatic installation system

A system has been developed at CERN for the automatic installation of CERNLIB sofware. This system currently runs on Unix systems only. It consists of the following components.

`libserv` Script that permits a developer or installer to request the installation of a new source file, the compilation of a program or both. For example **libserv put fatmen.car** will install the file **fatmen.car** in the **NEW** area on asis and on the **NEWPAMS** disk on CERNVM (a copy is also made in CMZ format).

`nfscp` This script is invoked when a client issues a **libserv put** command and copies the requested file to the target directories.

`nfsmake` This script is invoked when a client requests that a program is compiled, e.g. by **libserv compile kernlib**.

`asisserv` This is a slave script to **nfsmake** that is invoked on the target machines.

F.1 nfsmake

The *nfsmake* script can be used to rebuild all or parts of the CERN Program Library on one or more machines. By default, the library will be rebuilt on a list of machines that is defined in the script itself. At the time of writing, the library is built on the following platforms by default.

`apo`	Apollo 68000 architecture (node a-cernli)
`a10`	Apollo DN10000 architecture (node apofddi2)
`dec`	DECstation (MIPS chip) (node dspaw)
`hpx`	HP/UX (node hpcernlib)
`csf`	HP/UX (node csf)
`parc`	IBM RS6000 (node parcb)
`irs`	IBM RS6000 (node rscnas01)
`irx`	See below
`sgi`	Silicon Graphics (nodes shift1 and shift9)
`lnx`	IBM PC running LINUX operating system (node pcuslib)
`sun`	SUN (node sunpaw)
`osf`	Alpha OSF (node afcern)

New machines can be added trivially.

Appendix G: The CERNLIB *reference* platforms

The CERN Program Library is performed on the following *reference platforms*. With the current exception of the OSF and SGI machine, these machines are set up so that **/cern** is a link to **/afs/cern.ch/asis/@sys/cern**.

`dec`	DECstation 5000/200 node DXREF
`hpx`	HP9000/715 node HPREF
`irs`	RS6000/320 node RSREF
`osf`	Alpha 3000/300L node AFREF
`sgi`	SGI INDY node SGIREF
`sol`	Sun/Solaris Sparc Classic 4/15 node SUNSOREF
`sun`	Sun/OS Sparc-2 node SUNOSREF
`axclib`	Alpha 3000-600 in the CNLAVC
`vsclib`	VAXstation 4000-90 in the CNLAVC

Appendix H: Accessing the asis server at CERN

N.B. use *asisftp.cern.ch* when accessing asis as an ftp server and *asisnfs.cern.ch* when accessing asis as an NFS server. Originally, both of these services were offered on asis01 and old documentation may refer to this address.

H.1 Accessing the asis server as a file repository

You may access the **asis** server via anonymous ftp as show below.

Accessing the asis server via anonymous ftp

```
ftp asisftp.cern.ch      (IP address 128.141.202.89, userid anonymous)
Connected to asisftp.cern.ch.
220 asisftp FTP server (Version 2.0WU(14) Fri Sep 17 15:39:37 MET DST 1993) ready
.
Name (asisftp:jamie): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230-                  ----------------------------------------
230-                  Application Software Installation Server
230-                  ----------------------------------------
230-
230-   Welcome to the ASIS ftp server, developed by the CERN Computing and
230-   Networking Division to serve the High Energy Physics research community.
230-
230-   ftp clients may abort due to improper handling of such introductory
230-   messages. A dash (-) as the first character of your pw will suppress it.
230-
230-   The CERNlib software, located in the "cernlib" directory, is covered by
230-   CERN copyright. Before taking any material from this directory, please
230-   read the copyright notice "cernlib/copyright".
230-
230-   Please contact cernlib@cernvm.cern.ch for site registration. General
230-   support questions should be addressed to asis-support@asis01.cern.ch.
230-
230 Guest login ok, access restrictions apply.
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 72
-rw-r--r--   1 cernlib  local        5991 Apr  7  1993 README.cernlib
d--x--x--x   2 root     local         512 Sep 23 15:30 bin
drwxr-s---   2 cernlib  local        1024 Nov  2 15:07 cernlib
-rw-r--r--   1 cernlib  local        1490 Nov 13  1992 cernlib.registration
-rw-r--r--   1 asis     software    10107 Jun 15 09:59 cnasdoc.faq
drwxr-xr-x   7 cernlib  local         512 Oct 22 16:39 cnl
drwxr-xr-x   2 root     daemon        512 Sep 17 10:08 dev
d--x--x--x   2 root     local         512 Oct 26  1992 etc
drwxrwxr-x   2 defert   software      512 Jul  2 13:04 hepix
drwxrwxr-x 255 eric     staff        8704 Nov 16 12:22 preprints
drwxr-x---  20 asis     local         512 Sep 17 13:53 pub
dr-xr-xr-x   4 root     local         512 Sep 17 10:07 usr
drwxr-x---   2 asis     local         512 Sep 17 16:57 usr.local
226 Transfer complete.
```

```
ftp> cd cernlib
250-Please read the file README
250-  it was last modified on Wed Oct 27 17:32:33 1993 - 21 days ago
250 CWD command successful.
ftp>get README
200 PORT command successful.
150 Opening ASCII mode data connection for README (1288 bytes).
226 Transfer complete.
1307 bytes received in 0.006299 seconds (202.6 Kbytes/s)
ftp> ls
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
next_mach21
next_mach20
hp700_ux807
rs_aix32
pmax_ul4
sun4c_411
pc_dos50
sg_irix40
doc
copyright
rules
dec_ultrix4
README
alpha_osf1
apo_sr10
a10_sr10
hp700_ux90
share
sun4_solaris2
convex_os10
cray_unicos6
pc_linux
pro.src.car
new.src.car
mac_os6
alpha_wnt
sg_irix51
226 Transfer complete.
ftp>
```

You will only be able to access the **cernlib** directory if you are registered. If you cannot access the **cernlib** directory, follow the procedure below.

H.2 Registering for the asis server

If you are not registered you will not be able to access the cernlib directory. In this case retrieve the **cernlib.registration** file, fill it and return it via electronic mail. Access will normally be enabled within one working day.

Obtaining the CERNLIB registration form

```
ftp> get cernlib.registration
200 PORT command successful.
150 Opening ASCII mode data connection for cernlib.registration (1490 bytes).
226 Transfer complete.
1532 bytes received in 0.01752 seconds (85.37 Kbytes/s)
ftp> get README.cernlib
200 PORT command successful.
150 Opening ASCII mode data connection for README.cernlib (5991 bytes).
226 Transfer complete.
6146 bytes received in 0.02635 seconds (227.8 Kbytes/s)
ftp> quit
221 Goodbye
```

The cernlib registration form

```
              Access to asisftp.cern.ch (anonymous ftp)
                        Registration Form
-------------------------------------------------------------------------
This form is used to update our database on CERNlib users. This allows us
to keep you informed of various CERNlib issues and follow up requests for
software and documentation.

Please answer ALL of the questions. In particular, it is extremely useful
for us to know which systems and packages are of most importance.

Please send this information back by e-mail to cernlib@cern.ch.

Thank you in advance, CERN Program Library Office
-------------------------------------------------------------------------

 Name ...........:
 Email address ..:

 Postal Address .:



 Phone number ...:
 Fax number .....:                      (if available)

 Machine(s) to be enabled to access CERNlib software (please justify if
 you indicate more than one)
                          Machine1         Machine2     ....
 Internet name ..:
 IP address .....:
 Workstation type:
 Access by (1) ..:

    (1) please indicate the main role of the specified systems, e.g.
        personal workstation, group server, deparmental system etc.)
        registered (personal station, group system, university system,etc)
 Research area ..:
  (CERN experiment, HEP, chemistry, medicine, engineering, etc.)
```

```
Which programs (all CERNlib, fragments - which, public domain - which,
etc) and versions ( Sun, HP, etc) will you be taking regularly?
```

CERNLIB README file

```
This information file concerns the distribution, use and installation of
the CERN program library with particular emphasis on its availability
via the asis server.

Legal and commercial regulations covering the usage of this library are
described in the file "cernlib/copyright", the contents of which you
shall be deemed to have taken note.

To get access to the library material, users must be registered. The
procedure to follow is described in the file cernlib.registration,
which is also in this directory. Proceed as follows-

 cd /tmp
  ftp asisftp.cern.ch
 (IP address 128.141.202.89, userid anonymous)
 ftp> get cernlib.registration
 ftp> quit
* fill it and send it back to cernlib@cernvm.cern.ch
* when the request has been processed you will be notified by e-mail.


ORGANISATION
------------
On asis the library organisation is a tree structure as follows:

  README.cernlib   cernlib    cernlib.registration   .........
      |
      |
      -------------------------------------------
      |            |            |         |  |   |
 dec_ultrix4  hp700_ux807  sun4c_411   .......  transarc names which try
      |                                to indicate the hardware type and
      |                                operating system, e.g dec_ultrix4
      |                                for Decstation with Ultrix 4.x
      |
    ------------------------------------------------------------------
    |    |    |    |     |      |       |    |     |      |      |   |
   pro  new  old  cmz  patchy  mad  ........ distribution levels and
         |                                    products directories
 |
    --------------------------
    |    |    |    |    |    |
   tar  bin  lib  src  mgr  doc

where the directories contain

tar    Compressed tar files for efficient storage and transfer
bin    Ready to run modules, eg paw, paw++, kxterm, zftp, etc...
lib    Object libraries (ar format), eg packlib, graflib, etc...
src    Source files in Patchy format (card and cradles)
mgr    Tools and files for installers
doc    Documentation, mostly for Monte-Carlo libraries (mclibs),
       for other documentation of library material, see the /cernlib/doc
       tree.
```

```
DISTRIBUTION BY FTP
-------------------
The fastest and most convenient way to get parts or all of the library
products is by anonymous ftp to asisftp. If you want have the whole
distribution, complete sub-directories or packages, the best method is
to transfer the relevant compressed tar files and run the plitar shell
script to unpack them.  The compression factor is approximately 50%,
giving a substantial reduction in network traffic and transfer time.

To transfer a small number of specific products, go to the appropriate
directory to get the files you want. But you must be aware that in many
cases to run one module, you may need other files or modules too.

Examples
--------
1) To get the whole current production (pro) version of the Cern Program
   Library for an HP:

cd /tmp              (or the directory where you want to temporarily store
                      the compressed tar files)
ftp asisftp.cern.ch        (IP address 128.141.202.89)
    (give userid anonymous and password your e-mail address at the ftp prompts)
ftp> cd cernlib/hp700_ux807/pro/tar
ftp> get plitar
ftp> mget README*
ftp> mget *.contents
ftp> binary
ftp> mget *.Z
ftp> quit

Then
for csh do:
     setenv CERN /cern  (or where you want the files to be unpacked)
     setenv PLITMP /tmp (or the directory where you stored the tar files)
for sh/ksh do:
     CERN=/cern;export CERN     (see comments for csh)
     PLITMP=/tmp;export PLITMP  (see comments for csh)
     plitar xvf
will uncompress and untar the files into the specified directory.
More details of plitar are given below.


2) To get the current production (pro) versions of pawX11, paw++ and kxterm
   for a Decstation running Ultrix;

ftp> cd cernlib/dec_ultrix4/pro/bin
ftp> binary
ftp> get paw          (a shell script to invoke pawX11)
ftp> get pawX11       (the X11 version)
ftp> get paw++        (Motif version, needs X11 Release 4  and Motif 1.1)
ftp> get kxterm       (xterm handler for paw++, must be in the search PATH)
ftp> quit


Notes on the ftp access
-----------------------
For a variety of reasons (including ease of access, security, disk
```

```
space) many of the files and directories are reached by symbolic links.
This has the unfortunate side effect that you may lose your way trying
to go back using the ../ method until you get used to the tree. When in
doubt,
  cd /cernlib
will return you to the top of the tree.

At present, there are a limited number of ftp connections and at busy
periods you might be refused. We hope to improve this soon.


Notes on installing the tar files on your machine
-------------------------------------------------
- Put the plitar script in a convenient place ( e.g. $HOME/bin ) and make
  it executable (chmod +x plitar).
- Run plitar with the needed parameters, e.g.:
    plitar tvf          to verify the contents of the downloaded files
    plitar xvf cernlib  to unpack the tar files for the cern
                        library directory for example.

The plitar command makes use of two/three environment variables:

      Variable   Default   defining
      ---------- -------   --------------------
  -   CERN       /cern     target directory
  -   PLITMP     /tmp      location of tar files
( -   PLIUWC               obsolete since 93b, was type of machine before. )

Any can be redefined using setenv (in C-shell) or export (in sh,ksh)
For example:

      setenv PLITMP $HOME/tmp    or      export PLITMP=$HOME/tmp

to redefine the area where you installed the tar files (maybe because
/tmp is too small)

DISTRIBUTION on TAPE
--------------------
Tapes with the complete distribution in compressed tar format can be
ordered from the CERN Program Library Office, email address
 cernlib@cernvm.cern.ch
who will arrange the formalities.
```

H.3 Accessing the asis server as a file server

Appendix I: Description of the Unix scripts

I.1 cernsys and cernsys.csh

Shell scripts to set various environmental variables used for the installation of CERNLIB. The variables set are as follows:

`PLIUWC`	Unix Workstation Code
`PLISYS`	Unix system type
`PLINAME`	Variable used by **yexpand** to select appropriate installation options. e.g.

`Sun`	Sun
`Sun running Solaris`	Sun,Solaris
`RS6000`	IBMRT

The current definitions of the Unix workstation codes and system types are given below.

`alt`	Alliant
`amx`	Amiga-UX 3000
`apo`	Apollo 3000
`a10`	Apollo 10000
`cru`	Cray/UNICOS
`cvx`	Convex
`dec`	DECstation 5000
`hpx`	HP 9000/7xx
`irs`	IBM RISC 6000
`ibx`	IBM AIX/370
`lnx`	i486/Linux
`nsx`	NEC SX-3
`sgi`	Silicon Graphics
`sol`	Sun Solaris
`sun`	Sun SunOS

I.2 getdev

Shell script to get the device on which the directory given as first argument resides.

I.3 grouplib

I.4 makefile

I.5 makepack

makepack is the primary script for building the program library. It has the following options:

`-c` Run *compile* step
`-l` Run *library* step, i.e. build archive library

- `-p` Run *PATCHY* step. This step also runs the **KUIP** compiler **KUIPC** on any output files with extension **.cdf**.
- `-s` Run *split* step using **FCASPLIT**.
- `-i` Run *inlib* step to created installed libraries.
- `-C` Run special compilation step for generating installed libraries.
- `-t` Turn timing on for each step
- `-D` Development mode - each routine is compared against previous version and only changed routines are compiled.
- `-L` To add additional libraries to link step for target module.
- `-P` Production mode - rebuild all routines.

I.6 namefind

Shell script to extract information from a names file, e.g. **cernlib.names**.

I.7 plidd

I.8 plienv.csh and plienv.sh

Shell scripts for the C shell and Bourne/Korn shells to set the CERNLIB installation environment.

I.9 plilog

Shell script to view a CERNLIB installation logfile.

I.10 plitar

Shell script to unpack a CERNLIB (compressed) tar file.

I.11 shexit

shexit is a shell script to print a warning message in a standard fashion in case of abnormal termination of one of the installation scripts.

I.12 sumlog, sumlog2 and sumlog3

sumlog, **sumlog2** and **sumlog3** are shell scripts to summarize logfiles from installation jobs looking for potential problems. They use various control files, such as **sumlog.grep**, **sumlog.sed**, **sumlog.sed2**.

I.13 testpack

I.14 xdiff

xdiff is a shell script to compare complete directories.

I.15 xmv

xmv is a shell script to *mv* a set of files.

I.16 xvi

xvi is a shell script to run the *vi* editor in a separate window as a subprocess.

I.17 yexpand

YEXPAND is a shell script to expand environmental variables in **PATCHY** cradles.

Appendix J: Description the VM/CMS EXECs and service machines

Appendix K: Description of the VMS DCL command procedures

K.1 BACKUP_CERNLIB

Command file to build a **BACKUP** saveset containing material from the CERN:[PRO.DOC], [PRO.LIB], [PRO.EXE], [PRO.MGR], [PRO.SRC.COM] directories. The saveset is written to CERN:[PRO.BCK]CRNLIB.BCK. [1]

K.2 BACKUP_CMZ

Command file to build a **BACKUP** saveset of the current CERN:[PRO.CMZ] tree. The saveset is written to CERN:[PRO.BCK]CRNCMZ.BCK.

K.3 BACKUP_PATCHY

Command file to build a **BACKUP** saveset of the current PATCHY PRO tree. The saveset is written to CERN:[PRO.BCK]CRNPAT.BCK.

K.4 cernstart

CERNSTART.COM is a command file that is invoked at system startup time to perform functions such as

- Define system logical names
- Install software using the VMS **INSTALL** utility
- Perform NFS mounts
- . . .

N.B. The command file on the VXCERN cluster is CERN specific but may be used as an example for other systems.

K.5 enable_staging

This command file is part of the **VAXTAP** package [16] and is not required unless you wish to use the **STAGE** component of **VAXTAP**.

K.6 make

MAKE.COM is a partial emulation of the Unix *make* command. Together with **MAKEPACK.COM**, it is used to build the various components of the CERN Program Library.

The syntax is given below:

```
make [-d] [-f] [-i] [-l] [-n] [-s] [-t] target1 ... targetn
```

The options are as follows:

[1] **N.B. Note that VAX and Alpha specific savesets are to be found in the `CERNVAX` and `CERNAXP` trees respectively.**

`-d` Debug mode
`-f` Define makefile (dummy)
`-i` Interactive run
`-l` Library only
`-n` Noexecute mode - implies -i
`-s` Run job as subprocess (default)
`-t` Testpack flag

K.7 makepack

This command file is auxiliary to **MAKE.COM** and is used to build components of the CERN Library.

K.8 nfsdir

This command file can be used to sort the output of a directory command in a manner similar to the Unix `ls -ltr` command.

Example of using the NFSDIR command file

```
vxcrna:/cernlib > @cern:[new.mgr]nfsdir cern:[new.src.car]kern*.car

Total of 40 files, 9383/9383 blocks.
Directory CERN:[NEW.SRC.CAR]
KERNCDC.CAR;1           800/800      7-SEP-1988 20:33:44.00
KERNHYW.CAR;1            91/91      21-DEC-1989 12:41:02.00
KERNUNI.CAR;1           574/574     21-DEC-1989 12:41:45.00
KERNNOR.CAR;1           146/146     21-DEC-1989 16:03:21.00
KERNTMO.CAR;1            35/35      21-DEC-1989 16:03:41.00
KERNPDP.CAR;1           257/257      3-MAY-1990 02:24:10.00
KERNDGE.CAR;1            63/63       3-MAY-1990 02:24:26.00
KERNAMX.CAR;1            10/10      11-JUL-1991 04:38:43.00
KERNCVX.CAR;1           119/119     16-AUG-1991 16:08:48.00
KERNALI.CAR;1            23/23       7-OCT-1991 22:29:35.00
KERNHPX.CAR;1            14/14      22-MAY-1992 16:49:29.00
KERNIBX.CAR;1           326/326     22-MAY-1992 16:49:31.00
KERNA10.CAR;1           159/159      6-OCT-1992 12:02:09.00
KERNAPO.CAR;1           159/159      6-OCT-1992 12:02:09.00
KERNMPW.CAR;1            21/21       9-OCT-1992 01:55:41.00
KERNCRY.CAR;1           117/117     20-OCT-1992 18:29:12.00
KERNCRU.CAR;1           117/117     20-OCT-1992 18:29:12.00
KERNSGI.CAR;1            13/13      20-JAN-1993 12:39:13.00
KERNNXT.CAR;1            31/31      29-JAN-1993 12:02:29.00
KERNIBM.CAR;1           653/653     10-FEB-1993 00:38:04.00
KERNCMS.CAR;1           402/402     27-FEB-1993 02:24:52.00
KERNOS9.CAR;1            14/14       4-MAY-1993 21:35:21.00
KERNALT.CAR;1            61/61       4-JUN-1993 17:38:26.00
KERNDEC.CAR;1            38/38       4-JUN-1993 17:38:27.00
KERNOSF.CAR;1            38/38       4-JUN-1993 17:38:27.00
KERNVMI.CAR;1            38/38       4-JUN-1993 17:38:27.00
KERNDOS.CAR;1           153/153     23-JUL-1993 20:55:15.00
KERNGEN.CAR;1             4/4       12-AUG-1993 11:41:41.00
KERNNUM.CAR;1          1834/1834    25-AUG-1993 18:47:04.00
```

```
KERNNUMT.CAR;1          631/631     25-AUG-1993 18:48:28.00
KERNLNX.CAR;1            56/56       6-SEP-1993 15:34:22.00
KERNVAX.CAR;1           385/385     10-NOV-1993 10:09:46.33
KERNSOL.CAR;1             1/1       14-JAN-1994 15:48:56.77
KERNSUN.CAR;1             1/1       14-JAN-1994 15:48:56.77
KERNIRS.CAR;1            56/56      17-JAN-1994 17:16:11.98
KERNIRT.CAR;1            56/56      17-JAN-1994 17:16:11.98
KERNBIT.CAR;1           637/637     18-JAN-1994 18:43:03.85
KERNFOR.CAR;1           929/929     21-JAN-1994 19:56:29.15
KERNGENT.CAR;1          308/308     21-JAN-1994 19:57:34.87
```

K.9 plienv

This command file is used to define the environment expected for installation of the libraries. This includes the definition of various logical names and symbols.

Of particular importance are the following:

`PLINAME`	Symbol used by the installation jobs to select the correct version of the various packages. See section A for more details.
`MAKE`	Runs CERN_ROOT:[EXE]MAKE.COM
`MAKEPACK`	Runs CERN_ROOT:[EXE]MAKEPACK.COM
`TESTPACK`	Runs CERN_ROOT:[EXE]TESTPACK.COM
`PLIENV`	Runs CERN_ROOT:[EXE]PLIENV.COM
`PLILOG`	Runs CERN_ROOT:[EXE]PLILOG.COM
`SAY`	Defined as `WRITE SYS$OUTPUT`
`XTYPE`	Runs CERN_ROOT:[EXE]XTYPE.COM

K.10 plilog

Command file to edit a specified installation log file.

K.11 release

Command file to issue the appropriate **SET FILE/ENTER** and **SET FILE/REMOVE** commands to equivalence specific versions of the library with **OLD**, **PRO** and **NEW**. See section 7.2.1 for more information on this VXCERN specific command file.

K.12 stagelist

This command file is part of the **VAXTAP** package [16] and is not required unless you wish to use the **STAGE** component of **VAXTAP**.

K.13 setenv

Command file to define a global symbol or a logical name.

Examples of using the SETENV command

```
setenv baggins frodo

show symbol baggins

BAGGINS == "FRODO"

setenv -l baggins frodo

show logical baggins

   "BAGGINS" = "[.FRODO]" (LNM$PROCESS_TABLE)

setenv display zfatal:0

show display

   Device:    WS404:  [super]
   Node:      ZFATAL:0
   Transport: TCPIP
   Server:    0
   Screen:    0
```

K.14 testpack

Command file auxiliary to **MAKE** to build the test jobs for the various CERNLIB components.

K.15 ytocmz

Command file to convert a **PATCHY** card file into **CMZ** format.

Appendix L: Adding a new package to CERNLIB

The following information is normally only required by members of the CERN Program Library team.

To add a new package to CERNLIB, the following steps must be performed.

L.1 Unix systems

L.2 VMS systems

The file **CERN_ROOT:[MGR]MAKE.COM** must be modified to add the appropriate entries.

L.2.1 Adding a new stand-alone library

This is normally required when a new version of a Monte-Carlo generator is released. All that is required is that the appropriate source file (.CAR) and cradle (.CRA) are installed in the /cern/new/src/car area.

L.2.2 Adding a new CERN module

A new CERN module is added by updating the entry **l_cernpgm**, as shown below.

Updating the list of CERN modules

```
$! Initial configuration
$!
$ l_cernpgm ="SYSREQ  ,FATMEN  ,ZFTP    ,ZSERV   ,PAWSERV ,VAXTAP  ,"+ -
             "FLOP    ,TREE    ,CERNLIB ,FCASPLIT,XBANNER ,WYLBUR  ,"+ -
             "HIGZCONV,F2H     ,FCONV   "+ -
             l_packpgm+l_products

$! With ASTUCE added
$!
$ l_cernpgm ="SYSREQ  ,FATMEN  ,ZFTP    ,ZSERV   ,PAWSERV ,VAXTAP  ,"+ -
             "FLOP    ,TREE    ,CERNLIB ,FCASPLIT,XBANNER ,WYLBUR  ,"+ -
             "HIGZCONV,F2H     ,FCONV   ,ASTUCE  "+ -
             l_packpgm+l_products
```

Finally, the appropriate cradle (ASTUCE.CRA) is added to /cern/new/src/car.

L.2.3 Adding a new *user* module

A new user module is added in a similar manner, except that the list **l_userpgm** is updated (and the appropriate cradle added to /cern/new/src/car).

L.2.4 Adding a new module to an existing *set*

Let us take the example of the **HEPDB** modules. These are defined by the entry **hepdbset**.

Initial definition of hepdbset

```
$ l_hepdbset="CDSERV  ,HEPDB   "
```

The above definition will cause the modules **CDSERV** (the HEPDB server) and **HEPDB** (the interactive interface) to be built. We now wish to add two new modules: **CDMAKE**, to create new database files, and **CDMOVE**, the module responsibile for distribution of journal files between different servers.

This is done as follows:

1. Update the definition of **hepdbset**.
2. Add the appropriate *cradles* to the **/cern/new/src/car** area.

The definition of **hepdbset** is now as follows:

New definition of hepdbset

```
$ l_hepdbset="CDSERV  ,HEPDB   ,CDMOVE  ,CDMAKE  "
```

L.2.5 Adding a new package to PACKLIB

If we suppose that the initial definition of PACKLIB is as follows:

Initial definition of PACKLIB

```
$ l_packlib ="CSPACK  ,EPIO    ,FATLIB  ,FFREAD  ,HBOOK   ,KAPACK  ,"+ -
             "KUIP    ,MINUIT  ,ZBOOK   ,ZEBRA   ,CDLIB   "
```

we can add a new component simply by updating this list and adding the appropriate cradle.

L.3 VM/CMS systems

On VM/CMS systems, we must modify the file **MAKELIB NAMES** as appropriate.

L.3.1 Adding a new module to an existing *set*

In the following example we wish to add two new modules to the set **HEPDBSET**. This is initially defined as shown below.

Initial definition of HEPDBSET

```
* CERNPGM subentries      ---------------------------------------------

:NICK.MAKE_FATSET
:JOB.FATSET    :BOPT.TIME  0:00 JOBID FATSET
               :QSUB.
:FROM.FATMEN FATNEW FATSRV FATSEND

:NICK.MAKE_HEPDBSET
:JOB.HEPDBSET  :BOPT.TIME  0:00 JOBID HEPDBSET
               :QSUB.
:FROM.CDSERV HEPDB

:NICK.MAKE_RZCONV
:JOB.RZCONV    :BOPT.TIME  0:00 JOBID RZCONV
               :QSUB.
:FROM.RTOX RTOA RFRX RFRA
```

We now add the modules **CDMAKE** and **CDMOVE**, which gives us the following entry.

New definition of HEPDBSET

```
* CERNPGM subentries      ---------------------------------------------

:NICK.MAKE_FATSET
:JOB.FATSET    :BOPT.TIME  0:00 JOBID FATSET
               :QSUB.
:FROM.FATMEN FATNEW FATSRV FATSEND

:NICK.MAKE_HEPDBSET
:JOB.HEPDBSET  :BOPT.TIME  0:00 JOBID HEPDBSET
               :QSUB.
:FROM.CDSERV HEPDB CDMOVE CDMAKE

:NICK.MAKE_RZCONV
:JOB.RZCONV    :BOPT.TIME  0:00 JOBID RZCONV
               :QSUB.
:FROM.RTOX RTOA RFRX RFRA
```

In addition, we must add entries for **CDMOVE** and **CDMAKE** as shown below.

Build definitions for CDMAKE and CDMOVE

```
:NICK.MAKE_CDMAKE
:JOB.CDMAKE    :BOPT.TIME  1:00 JOBID CDMAKE
               :QSUB.
:HASM.(BATCH   :FORTVS.(TERM TRMFLG FLAG(I)
:CERNLIB.PACKLIB ( LINK
:LOAD.CDMAKE  ( NOAUTO CLEAR
:INCLUDE.ENDMODU       :GENMOD.CDMAKE  ( TO ENDMODU RMODE ANY AMODE ANY
```

```
:NICK.MAKE_CDMOVE
:JOB.CDMOVE    :BOPT.TIME  1:00 JOBID CDMOVE
               :QSUB.
:HASM.(BATCH   :FORTVS.(TERM TRMFLG FLAG(I)
:CERNLIB.PACKLIB ( LINK
:LOAD.CDMOVE  ( NOAUTO CLEAR
:INCLUDE.ENDMODU       :GENMOD.CDMOVE  ( TO ENDMODU RMODE ANY AMODE ANY
```

Appendix M: Changing the version of an existing package

Certain packages, notably the Monte Carlo libraries, contain the version number in the source and library file names. When a new version is received from the authors, the following must be done:

- The new source file must be installed on asis.
- A new cradle must be created, normally by copying the old one, e.g. cp jetset73.cra jetset74.cra.
- The *make* files must be updated appropriately.

M.1 VM/CMS

On VM/CMS systems, the following must be performed:

- A new entry must be made in **MAKELIB NAMES** on the group disk.
- The **CERNLIB** exec and **PLIENV** exec must be changed, if the new version is to become the default.

M.2 VMS

On VMS systems, the **CERNLIB** command must be modified and rebuilt, if the new version is to become the default.

M.3 Unix

On Unix systems, the **CERNLIB** script must be modified and rebuilt, if the new version is to become the default.

Appendix N: Testing

Test jobs are generally contained in the source file of the package that they are testing and are extracted using a cradle composed of the package name appended by the letter T, e.g. **hbookt]** for HBOOK. The tests for **KERNGEN** are contained in a separate PAM file, KERNGENT.

N.1 VM

The tests are run using e.g. **SLIB MAKE TEST_HBOOK**

N.2 VMS

The tests are run using e.g. **TESTPACK HBOOK**

N.3 Unix

The tests are run using e.g. **TESTPACK HBOOK**

N.4 List of tests

```
BVSL
COJETS
EPT1L
EPT1S
EPT2L
EPT2S
EPT3L
EPT3S
EPT4L
EPT4S
EURODEC
FFREAD
FLOP
FORCCR
FRITIOF
G321X1
G321X2
G321X3
G321X4
G321X5
GARFIELD
HBOOK
hptGDDM
hptGKS
hptGL
```

```
hptX11
hptgks
hptx11
hztGDDM
hztGKS
hztGL
hztX11
hztgks
hztx11
ISAJET
JETSET74
KAPACK
KERNASW
KERNGEN
KERNNUM
KUIPC
LATTCR
LEPTO
MINUIT
MPA
PAW
PDFLIB
PHOTOS
POISCR
TRIPCR
ZBOOK
zebfc1
zebfc2
zebfc3
zebfz1
zebfz2
zebfz3
zebfz4
zebfz5
zebfz6
zebfz7
zebfz8
zebfz9
zebjz1
zebmz1
zebrz1
zebrz2
zebtlib
```

Appendix O: Making a release of the CERN Program Library

N.B. this information is for CERN Program Library staff only

O.1 VM/CMS

Once the files have been built in the **NEW** area, they can be released using the procedure **SWAPLIB EXEC**.

N.B. This procedure must be updated for each release, e.g. for Monte Carlo version numbers.

Before running the **SWAPLIB** procedure, limit the potential disruption to users by requesting that the operators stop the batch queues and all service machines.

Each area (OLD, PRO, NEW) uses 5 disks. The starting addresses for the different areas are 300, 310 and 320 respectively.

300 O/COM - 1 cylinder
301 O/CAR - 109 cylinders
302 O/OBJ - 109 cylinders
303 O/LIB - 250 cylinders
304 O/CMZ - 109 cylinders

This procedure performs the following steps:

1. Accesses all disks in RW mode
2. Changes the disk labels (e.g. N/CAR to P/CAR)
3. Erases all files from the OLD area
4. Copies the current NEW area to the disks freed by the previous step. *This is now done using DFSMS, which unfortunately overwrites the disk label - see later.*
5. Renames files, e.g. PACKNEW to PACKLIB, N_PAWX11 to PAWX11 etc.
6. Fixes the names for certain files, e.g. YPATCH$M.
7. Swaps the disk addresses. **N.B. The best way to swap the disks is to use DIRMAINT, as described below**.
8. Sends a warning message to all users.

Once this has been done, perform the following:

- Check the **CONSOLE LOG** for errors
- Check the contents of the PRO disk
- Change the **GIME NOTICE** files and disk labels on the new NEW disks.

O.1.1 Using DIRMAINT to swap the disk addresses

This procedure is error prone and so should be used with caution.

1. Change the **303** entry to **323**
2. Change the **313** entry to **303**
3. Change the initial **323** entry to **313**

The same can be done using **SPACE CHNGDISK**, as is done by **SWAPLIB**, but this takes more time.

The CONSOLE logs from the 94A release are included below. Note that the production disks were changed using DIRMAINT as described above.

Release version 94A on CERNVM

```
SWAPLIB

Step 1: Relabel disks
Type Go, Retry, Skip, Quit <DEF=G>

----------------------------------------------------------
Current Program Library disks:               Relabelled To:
----------------------------------------------------------
DMSFOR605R Enter disk label:
Disk CERNCRA =CERNLIB 0310: P/COM (Z 01C0)     O/COM
DMSFOR605R Enter disk label:
Disk CERNPAMS=CERNLIB 0311: P/CAR (X 0311)     O/CAR
DMSFOR605R Enter disk label:
Disk CERNTEXT=CERNLIB 0312: P/OBJ (W 0312)     O/OBJ
DMSFOR605R Enter disk label:
Disk CERNLIBS=CERNLIB 0313: /Q/CRN (V 01C1)     O/LIB
DMSFOR605R Enter disk label:
Disk CERNCMZ =CERNLIB 0314: P/CMZ (U 0314)     O/CMZ
DMSFOR605R Enter disk label:
Disk OLDCRA  =CERNLIB 0300: O/COM (T 01C2)     N/COM
DMSFOR605R Enter disk label:
Disk OLDPAMS =CERNLIB 0301: O/CAR (R 0301)     N/CAR
DMSFOR605R Enter disk label:
Disk OLDTEXT =CERNLIB 0302: O/OBJ (O 0302)     N/OBJ
DMSFOR605R Enter disk label:
Disk OLDLIBS =CERNLIB 0303: O/LIB (N 01C3)     N/LIB
DMSFOR605R Enter disk label:
Disk OLDCMZ  =CERNLIB 0304: O/CMZ (M 0304)     N/CMZ
DMSFOR605R Enter disk label:
Disk NEWCRA  =CERNLIB 0320: N/COM (L 01C4)     P/COM
DMSFOR605R Enter disk label:
Disk NEWPAMS =CERNLIB 0321: N/CAR (K 0321)     P/CAR
DMSFOR605R Enter disk label:
Disk NEWTEXT =CERNLIB 0322: N/OBJ (J 0322)     P/OBJ
DMSFOR605R Enter disk label:
Disk NEWLIBS =CERNLIB 0323: N/LIB (I 01C5)     /Q/CRN
DMSFOR605R Enter disk label:
Disk NEWCMZ  =CERNLIB 0324: N/CMZ (H 0324)     P/CMZ
```

```
Step 2: Clear OLDCRA + Backup NEWCRA
Type Go, Retry, Skip, Quit <DEF=G>

1C2 replaces T (1C2)
UserID: CERNLIB   Date: 03/15/94  Time: 09:10:55
COPY     L        01C2     (        INFO     OLDDATE  STATUS
COPY     return code =   0
LABEL  VDEV M  STAT  CYL TYPE BLKSIZE  FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
P/COM  1C2  T   R/W    1 3390 4096        62        135-75         45        180

Step 3: Clear OLDPAMS + Backup NEWPAMS
Type Go, Retry, Skip, Quit <DEF=G>

301 replaces R (301)
UserID: CERNLIB   Date: 03/15/94  Time: 09:11:03
COPY     K        0301     (        INFO     OLDDATE  STATUS
Copying block    1800 of   19620
Copying block    3600 of   19620
Copying block    5400 of   19620
Copying block    7200 of   19620
Copying block    9000 of   19620
Copying block   10800 of   19620
Copying block   12600 of   19620
Copying block   14400 of   19620
Copying block   16200 of   19620
Copying block   18000 of   19620
COPY     return code =   0
LABEL  VDEV M  STAT  CYL TYPE BLKSIZE  FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
P/CAR  301  R   R/W  109 3390 4096       401      17834-91       1786      19620

Step 4: Clear OLDTEXT + Backup NEWTEXT
Type Go, Retry, Skip, Quit <DEF=G>

302 replaces O (302)
UserID: CERNLIB   Date: 03/15/94  Time: 09:12:19
COPY     J        0302     (        INFO     OLDDATE  STATUS
Copying block    1800 of   19620
Copying block    3600 of   19620
Copying block    5400 of   19620
Copying block    7200 of   19620
Copying block    9000 of   19620
Copying block   10800 of   19620
Copying block   12600 of   19620
Copying block   14400 of   19620
Copying block   16200 of   19620
Copying block   18000 of   19620
COPY     return code =   0
LABEL  VDEV M  STAT  CYL TYPE BLKSIZE  FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
P/OBJ  302  O   R/W  109 3390 4096        46       9013-46      10607      19620

Step 5: Clear OLDLIBS + Backup NEWLIBS
Type Go, Retry, Skip, Quit <DEF=G>

1C3 replaces N (1C3)
UserID: CERNLIB   Date: 03/15/94  Time: 09:13:34
COPY     I        01C3     (        INFO     OLDDATE  STATUS
Copying block    1800 of   45000
```

```
Copying block    3600 of   45000
Copying block    5400 of   45000
Copying block    7200 of   45000
Copying block    9000 of   45000
Copying block   10800 of   45000
Copying block   12600 of   45000
Copying block   14400 of   45000
Copying block   16200 of   45000
Copying block   18000 of   45000
Copying block   19800 of   45000
Copying block   21600 of   45000
Copying block   23400 of   45000
Copying block   25200 of   45000
Copying block   27000 of   45000
Copying block   28800 of   45000
Copying block   30600 of   45000
Copying block   32400 of   45000
Copying block   34200 of   45000
Copying block   36000 of   45000
Copying block   37800 of   45000
Copying block   39600 of   45000
Copying block   41400 of   45000
Copying block   43200 of   45000
COPY     return code =    0
LABEL  VDEV M  STAT   CYL TYPE BLKSIZE  FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
/Q/CRN 1C3  N   R/W  250 3390 4096       279      43518-97       1482      45000

Step 6: Clear OLDCMZ + Backup NEWCMZ
Type Go, Retry, Skip, Quit <DEF=G>

304 replaces M (304)
UserID: CERNLIB   Date: 03/15/94  Time: 09:15:59
COPY     H         0304     (        INFO     OLDDATE  STATUS
Copying block    1800 of   19620
Copying block    3600 of   19620
Copying block    5400 of   19620
Copying block    7200 of   19620
Copying block    9000 of   19620
Copying block   10800 of   19620
Copying block   12600 of   19620
Copying block   14400 of   19620
Copying block   16200 of   19620
Copying block   18000 of   19620
COPY     return code =    0
LABEL  VDEV M  STAT   CYL TYPE BLKSIZE  FILES  BLKS USED-(%) BLKS LEFT  BLK TOTAL
P/CMZ  304  M   R/W  109 3390 4096       123      15311-78       4309      19620

Step 7: Rename special files in NEWLIBS
Type Go, Retry, Skip, Quit <DEF=G>



f# Old file:                New file:            RC
-- --------------------   -------------------- --
1  VPACKNEW TXTLIB   I2   VPACKLIB =        =  0
2  VMATHNEW TXTLIB   I2   VMATHLIB =        =  0
3  VKERNNEW TXTLIB   I2   VKERNLIB =        =  0
```

```
 4  PDFNEW   TXTLIB   I2   PDFLIB   =         =  0
 5  PAWNEW   TXTLIB   I2   PAWLIB   =         =  0
 6  PACKNEW  TXTLIB   I2   PACKLIB  =         =  0
 7  MATHNEW  TXTLIB   I2   MATHLIB  =         =  0
 8  KERNNEW  TXTLIB   I2   KERNLIB  =         =  0
 9  GRAFNEW  TXTLIB   I2   GRAFLIB  =         =  0
10  N_HTONEW MODULE   I1   N_HTOLIB =         =  0
11  PACKNEW  ICAFILE  I2   PACKLIB  =         =  28
```

```
f# Old file:               New file:            RC
-- --------------------    -------------------- --
 1  N_TWISTE TXTLIB   I2   TWISTE   =         =  0
 2  N_PHTOOL TXTLIB   I2   PHTOOL   =         =  0
 3  N_PHOTOS TXTLIB   I2   PHOTOS   =         =  0
 4  N_LEPTO6 TXTLIB   I2   LEPTO6   =         =  0
 5  N_JETSET TXTLIB   I2   JETSET   =         =  0
 6  N_JETNET TXTLIB   I1   JETNET   =         =  0
 7  N_ISAJET TXTLIB   I2   ISAJET   =         =  0
 8  N_HERWIG TXTLIB   I2   HERWIG   =         =  0
 9  N_GRAFX1 TXTLIB   I2   GRAFX1   =         =  0
10  N_GRAFGK TXTLIB   I2   GRAFGK   =         =  0
11  N_GRAFGD TXTLIB   I2   GRAFGD   =         =  0
12  N_GEANT3 TXTLIB   I2   GEANT3   =         =  0
13  N_FRITIO TXTLIB   I2   FRITIO   =         =  0
14  N_EURODE TXTLIB   I2   EURODE   =         =  0
15  N_COJETS TXTLIB   I2   COJETS   =         =  0
116 N_CKERNE TXTLIB   I2   CKERNE   =         =  0
17  N_ARIADN TXTLIB   I2   ARIADN   =         =  0
18  N_ZSERV  MODULE   I2   ZSERV    =         =  0
19  N_ZFTP   MODULE   I2   ZFTP     =         =  0
20  N_YTOBIN MODULE   I2   YTOBIN   =         =  0
21  N_YTOBCD MODULE   I2   YTOBCD   =         =  0
22  N_YPATCH MODULE   I2   YPATCH   =         =  0
23  N_VGARF  MODULE   I1   VGARF    =         =  0
24  N_TREE   MODULE   I2   TREE     =         =  0
25  N_RTOX   MODULE   I2   RTOX     =         =  0
26  N_RTOA   MODULE   I2   RTOA     =         =  0
27  N_RFRX   MODULE   I2   RFRX     =         =  0
28  N_RFRA   MODULE   I2   RFRA     =         =  0
29  N_PAWX11 MODULE   I2   PAWX11   =         =  0
30  N_PAWSER MODULE   I2   PAWSER   =         =  0
31  N_PAWGKS MODULE   I2   PAWGKS   =         =  0
32  N_PAWGDD MODULE   I2   PAWGDD   =         =  0
33  N_MAKEDE MODULE   I2   MAKEDE   =         =  0
34  N_KUIPC  MODULE   I2   KUIPC    =         =  0
35  N_HTOLIB MODULE   I1   HTOLIB   =         =  0
36  N_HIGZCO MODULE   I2   HIGZCO   =         =  0
37  N_HEPDB  MODULE   I2   HEPDB    =         =  0
38  N_GRTPS  MODULE   I2   GRTPS    =         =  0
39  N_GRTGKS MODULE   I2   GRTGKS   =         =  0
40  N_GARF   MODULE   I2   GARF     =         =  0
41  N_FLOP   MODULE   I2   FLOP     =         =  0
42  N_FCASPL MODULE   I2   FCASPL   =         =  0
43  N_FATMEN MODULE   I2   FATMEN   =         =  0
44  N_DZEX11 MODULE   I2   DZEX11   =         =  0
```

```
45 N_DZEGKS MODULE   I2   DZEGKS   =        =  0
46 N_DZEGDD MODULE   I2   DZEGDD   =        =  0
47 N_CMZ    MODULE   I2   CMZ      =        =  0
48 N_CDSERV MODULE   I2   CDSERV   =        =  0
49 N_BZFTP  MODULE   I2   BZFTP    =        =  0
50 N_BANNER MODULE   I2   BANNER   =        =  0


f# Old file:              New file:            RC
-- --------------------   -------------------- --
1  VPACKLIB TXTLIB   V2   VPACKOLD TXTOLD   =  0
2  VMATHLIB TXTLIB   V2   VMATHOLD TXTOLD   =  0
3  VKERNLIB TXTLIB   V2   VKERNOLD TXTOLD   =  0
4  PDFLIB   TXTLIB   V2   PDFOLD   TXTOLD   =  0
5  PAWLIB   TXTLIB   V2   PAWOLD   TXTOLD   =  0
6  PACKLIB  TXTLIB   V2   PACKOLD  TXTOLD   =  0
7  MATHLIB  TXTLIB   V2   MATHOLD  TXTOLD   =  0
8  KERNLIB  TXTLIB   V2   KERNOLD  TXTOLD   =  0
9  GRAFLIB  TXTLIB   V2   GRAFOLD  TXTOLD   =  0


f# Old file:              New file:            RC
-- --------------------   -------------------- --
1  TWISTER  TXTLIB   V2   O_TWISTE =        =  0
2  PHTOOLS  TXTLIB   V2   O_PHTOOL =        =  0
3  PHOTOS   TXTLIB   V2   O_PHOTOS =        =  0
4  PDFNEW   TXTLIB   V0   O_PDFNEW =        =  0
5  LEPTO61  TXTLIB   V2   O_LEPTO6 =        =  0
6  JSET73   TXTLIB   V2   O_JSET73 =        =  0
7  JSET63   TXTLIB   V2   O_JSET63 =        =  0
8  JETNET   TXTLIB   V1   O_JETNET =        =  0
9  ISAJET72 TXTLIB   V2   O_ISAJET =        =  0
10 ISAJET65 TXTLIB   V2   O_ISAJET =        =  28
11 HERWIG56 TXTLIB   V2   O_HERWIG =        =  0
12 GRAFX11  TXTLIB   V2   O_GRAFX1 =        =  0
13 GRAFGKS  TXTLIB   V2   O_GRAFGK =        =  0
14 GRAFGDDM TXTLIB   V2   O_GRAFGD =        =  0
15 GEANT316 TXTLIB   V2   O_GEANT3 =        =  0
16 GEANT315 TXTLIB   V2   O_GEANT3 =        =  28
17 GEANT314 TXTLIB   V2   O_GEANT3 =        =  28
18 FRITIOF  TXTLIB   V2   O_FRITIO =        =  0
19 EURODEC  TXTLIB   V2   O_EURODE =        =  0
20 COJETS   TXTLIB   V2   O_COJETS =        =  0
21 CKERNEL  TXTLIB   V2   O_CKERNE =        =  0
22 ARIADNE  TXTLIB   V2   O_ARIADN =        =  0


f# Old file:              New file:            RC
-- --------------------   -------------------- --
1  VPACKOLD TXTOLD   V2   =        TXTLIB   =  0
2  VMATHOLD TXTOLD   V2   =        TXTLIB   =  0
3  VKERNOLD TXTOLD   V2   =        TXTLIB   =  0
4  PDFOLD   TXTOLD   V2   =        TXTLIB   =  0
5  PAWOLD   TXTOLD   V2   =        TXTLIB   =  0
```

```
 6  PACKOLD  TXTOLD   V2   =        TXTLIB   =  0
 7  MATHOLD  TXTOLD   V2   =        TXTLIB   =  0
 8  KERNOLD  TXTOLD   V2   =        TXTLIB   =  0
 9  GRAFOLD  TXTOLD   V2   =        TXTLIB   =  0



 f# Old file:                New file:             RC
 -- --------------------     -------------------- --
 1  ZSERV    MODULE   V2   O_ZSERV  =         =  0
 2  ZFTP     MODULE   V2   O_ZFTP   =         =  0
 3  YTOBIN$M MODULE   V2   O_YTOBIN =         =  0
 4  YTOBCD$M MODULE   V2   O_YTOBCD =         =  0
 5  YPATCH$M MODULE   V2   O_YPATCH =         =  0
 6  VGARF    MODULE   V1   O_VGARF  =         =  0
 7  TREE     MODULE   V2   O_TREE   =         =  0
 8  RTOX     MODULE   V2   O_RTOX   =         =  0
 9  RTOA     MODULE   V2   O_RTOA   =         =  0
 10 RFRX     MODULE   V2   O_RFRX   =         =  0
 11 RFRA     MODULE   V2   O_RFRA   =         =  0
 12 PAWX11   MODULE   V2   O_PAWX11 =         =  0
 13 PAWSERV  MODULE   V2   O_PAWSER =         =  0
 14 PAWGKS   MODULE   V2   O_PAWGKS =         =  0
 15 PAWGDDM  MODULE   V2   O_PAWGDD =         =  0
 16 MAKEDECK MODULE   V2   O_MAKEDE =         =  0
 17 KUIPC    MODULE   V2   O_KUIPC  =         =  0
 18 HTOLIB   MODULE   V1   O_HTOLIB =         =  0
 19 HIGZCONV MODULE   V2   O_HIGZCO =         =  0
 20 HEPDB    MODULE   V2   O_HEPDB  =         =  0
 21 GRTPS    MODULE   V2   O_GRTPS  =         =  0
 22 GRTGKS   MODULE   V2   O_GRTGKS =         =  0
 23 GARF     MODULE   V1   O_GARF   =         =  0
 24 FLOP     MODULE   V2   O_FLOP   =         =  0
 25 FCASPLIT MODULE   V2   O_FCASPL =         =  0
 26 FATMEN   MODULE   V2   O_FATMEN =         =  0
 27 DZEX11   MODULE   V2   O_DZEX11 =         =  0
 28 DZEGKS   MODULE   V2   O_DZEGKS =         =  0
 29 DZEGDDM  MODULE   V2   O_DZEGDD =         =  0
 30 CMZ      MODULE   V2   O_CMZ    =         =  0
 31 CDSERV   MODULE   V2   O_CDSERV =         =  0
 32 BZFTP    MODULE   V2   O_BZFTP  =         =  0
 33 BANNER   MODULE   V2   O_BANNER =         =  0


 Step 8: Cycle the disks
 Type Go, Retry, Skip, Quit <DEF=G>
 q
```

The library disks where then swapped by modifying the CP directory using DIRMAINT. Once the following messages have been received, SWAPLIB can continue (or one can reinvoke SWAPLIB and skip steps 1-7 as is shown below). b

DIRMAINT messages from swapping the library disks

```
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM REPLACE ).
DVHMCB009I Directory update ONLINE: Command DIRM REPLACE CERNLIB 42588936 .
```

Swapping the CERNLIB disks using SWAPLIB

```
Step 1: Relabel disks
Type Go, Retry, Skip, Quit <DEF=G>
s

Step 2: Clear OLDCRA + Backup NEWCRA
Type Go, Retry, Skip, Quit <DEF=G>
s

Step 3: Clear OLDPAMS + Backup NEWPAMS
Type Go, Retry, Skip, Quit <DEF=G>
s

Step 4: Clear OLDTEXT + Backup NEWTEXT
Type Go, Retry, Skip, Quit <DEF=G>
s

Step 5: Clear OLDLIBS + Backup NEWLIBS
Type Go, Retry, Skip, Quit <DEF=G>
s

Step 6: Clear OLDCMZ + Backup NEWCMZ
Type Go, Retry, Skip, Quit <DEF=G>
s

Step 7: Rename special files in NEWLIBS
Type Go, Retry, Skip, Quit <DEF=G>
s

Step 8: Cycle the disks
Type Go, Retry, Skip, Quit <DEF=G>


Swapping CERNLIB 304 TO 334
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 304 TO 334 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 304 TO 334 .

Swapping CERNLIB 303 TO 333
Type Go, Retry, Skip, Quit <DEF=G>
s

Swapping CERNLIB 302 TO 332
```

```
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 302 TO 332 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 301 TO 331
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 301 TO 331 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 302 TO 332 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 301 TO 331 .

Swapping CERNLIB 300 TO 330
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 300 TO 330 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 314 TO 304
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 314 TO 304 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB

Swapping CERNLIB 313 TO 303
Type Go, Retry, Skip, Quit <DEF=G>
s

Swapping CERNLIB 312 TO 302
Type Go, Retry, Skip, Quit <DEF=G>
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 300 TO 330 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 314 TO 304 .
HCPBER4123I Do not forget to BERU RESET, you are getting addicted...

SPACE : CHNGDISK CERNLIB 312 TO 302 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 311 TO 301
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 311 TO 301 received.
```

```
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB

Swapping CERNLIB 310 TO 300
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 310 TO 300 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 312 TO 302 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 324 TO 314
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 324 TO 314 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB

Swapping CERNLIB 323 TO 313
Type Go, Retry, Skip, Quit <DEF=G>
s

Swapping CERNLIB 322 TO 312
Type Go, Retry, Skip, Quit <DEF=G>
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 311 TO 301 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).
09:32:22  * MSG FROM JAMIE    : I'd like to link 'M' to CERNLIB 313 (Your 01C1)
09:32:22  * MSG FROM JAMIE    : Please detach and issue CP SMSG JAMIE OK or NO

DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 310 TO 300 .
SPACE : CHNGDISK CERNLIB 322 TO 312 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 321 TO 311
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 321 TO 311 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 324 TO 314 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).
HCPBER4123I Do not forget to BERU RESET, you are getting addicted...
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 322 TO 312 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 320 TO 310
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 320 TO 310 received.
```

```
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 321 TO 311 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 334 TO 324
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 334 TO 324 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB

Swapping CERNLIB 333 TO 323
Type Go, Retry, Skip, Quit <DEF=G>
s

Swapping CERNLIB 332 TO 322
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 332 TO 322 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 320 TO 310 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 334 TO 324 .
DVHDMA008I Source update applied, next ONLINE scheduled immediate (command
   DIRM CHVADDR ).

Swapping CERNLIB 331 TO 321
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 331 TO 321 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB

Swapping CERNLIB 330 TO 320
Type Go, Retry, Skip, Quit <DEF=G>

SPACE : CHNGDISK CERNLIB 330 TO 320 received.
SPACE : Request to CHNGDISK sent to DIRMAINT
SPACE : Wait for DIRMAINT to reply to CERNLIB

Step 9: Notify users
Type Go, Retry, Skip, Quit <DEF=G>

Enter the magic command...
beru uco b
Magic word ?
DVHMCB009I Directory update ONLINE: Command DIRM CHVADDR CERNLIB 332 TO 322 .

HCPBER4120I Ecce, in manu tua est berum, tamen animam illius serva.
09:37:07  * WNG FROM CERNLIB :   The Q-disk has been upgraded. Please type "NEWQDISK"
HCPBER4121I VM hominibus berum donavit, ergo jactare illius potest homo solus.
```

O.2 VMS

On **VXCERN**, each release has a directory tree, e.g. CERN:[94A]. The command **SET FILE/ENTER** is used to create aliases for the **OLD**, **PRO** and **NEW** releases.

The following command file, CERN:[PRO.MGR]RELEASE.COM, can be used to change the aliases. It must be updated for each release.

Note that it leaves **NEW** and **PRO** pointing to the same directory. A new *NEW* area is only created once **PRO** is stable.

RELEASE.COM

```
$!Release procedure for Vax/VMS
$
$ CERNDIR=F$TRNLNM("CERN")-".]"+"]"
$ set default 'CERNDIR'
$
$!set file/remove new.dir;*
$ set file/remove pro.dir;*
$ set file/remove old.dir;*
$
$!delete/log [.93b...]*.*.*
$!backup/log [.93d...] [.94a...]
$
$!set file/enter=new.dir 94b.dir
$ set file/enter=pro.dir 94a.dir
$ set file/enter=old.dir 93d.dir
$
$ set default [.cmz]
$
$!set file/remove new.dir;*
$ set file/remove pro.dir;*
$ set file/remove old.dir;*
$
$!delete/log [.1_43...]*.*.*
$!backup/log [.1_45...] [.1_46...]
$
$!set file/enter=new.dir 1_46.dir
$ set file/enter=pro.dir 1_45.dir
$ set file/enter=old.dir 1_44.dir
```

After changing the file pointers, the file CERN:[000000]RELEASE.LEVEL should also be modified to reflect the new situation.

Example RELEASE.LEVEL file

```
CERNLIB OLD=93D
CERNLIB PRO=94A
CERNLIB NEW=94B
CMZ     OLD=1.44
CMZ     PRO=1.45
CMZ     NEW=1.46
GKS     OLD=3.2
GKS     PRO=3.2
GKS     NEW=3.2
```

```
LAPACK  OLD=1.0
LAPACK  PRO=1.0B
LAPACK  NEW=1.0B
NAG     OLD=MARK11
NAG     PRO=MARK15
NAG     NEW=MARK15
PHIGS   OLD=V20
PHIGS   PRO=V21
PHIGS   NEW=V21
```

O.3 Unix

Appendix P: Access to licensed products distributed by CERN

P.1 CMZ

CMZ is the product of a software company (CodeMe S.A.R.L.); and it uses parts of the CERN Program Library. The following agreement has been reached between CERN and the supplier:

1. CERN users will have the right to use CMZ on all machines on the CERN site.
2. In CERN member states all institutions collaborating with CERN, national research laboratories in nuclear and particle physics and academic physics departments will receive CMZ for free.
3. Groups or individuals outside CERN member states collaborating with the experimental programme of CERN or of one of the institutions mentioned in the previous paragraph will have the right to receive CMZ free of charge and to use it for the work they are doing in the framework of the above collaboration. If they want to use CMZ for any other activity, then they must obtain a license from the supplier.
4. All cases not directly covered by the above rules 1,2 and 3 will have to be negotiated directly with the supplier. More information about licence fees may be obtained from CODEME@CERNVM.
5. The CERN agreement with the company includes maintenance and development coverage for users of CMZ in the first three categories listed above

Individuals or institutions entitled to receive CMZ free of charge according to the above conditions, should request the program and the documentation from the CERN Program Library Office.

P.2 GKS

P.3 GPHIGS

P.4 LAPACK

LAPACK may be obtained by anonymous ftp from netlib2.cs.utk.edu as shown below.

Obtaining the LAPACK tar file

```
zfatal:/home/cp/jamie (4) ftp netlib2.cs.utk.edu
Connected to netlib2.cs.utk.edu.
220 netlib2 FTP server (Version 2.1aWU(1) Thu Jun 3 23:00:04 EDT 1993) ready.
Name (netlib2.cs.utk.edu:jamie): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230 Guest login ok, access restrictions apply.
ftp> bin
200 Type set to I.
ftp> ls *.tar.z
200 PORT command successful.
150 Opening ASCII mode data connection for file list.
lapack.tar.z
manpages.tar.z
revisions1.0a.tar.z
```

```
revisions1.0b.tar.z
testing.tar.z
timing.tar.z
226 Transfer complete.
ftp> mget *

...
```

```
===================
LAPACK README FILE
===================

VERSION 1.0  :  February 29, 1992
VERSION 1.0a :  June 30, 1992
VERSION 1.0b :  October 31, 1992

DATE:  October 31, 1992
```

LAPACK is a library of Fortran 77 subroutines for solving the most commonly occurring problems in numerical linear algebra. It is public-domain software, and can be used freely.

The tar tape contains the Fortran source for LAPACK, the testing programs, and the timing programs.

It also contains Fortran code for the Basic Linear Algebra Subprograms (the Level 1, 2, and 3 BLAS) needed by LAPACK. However this code is intended for use only if there is no other implementation of the BLAS already available on your machine; the efficiency of LAPACK depends very much on the efficiency of the BLAS.

The complete package, including test code and timing programs in four different Fortran data types (real, complex, double precision, double complex), contains some 600,000 lines of Fortran source and comments. You will need approximately 28 Mbytes to read the complete tape. We recommend that you run the testing and timing programs. The total space requirements for the testing and timing for all four data types, including the object files, is approximately 70 Mbytes.

A README file containing the information in this letter and a QUICK_INSTALL file containing a quick reference guide to the installation process are located in the LAPACK directory. Postscript and LaTeX versions of the Installation Guide are in the LAPACK/INSTALL directory, in the files install.tex, psfig.tex, install.ps, and org2.ps. Consult the Installation Guide for further details on installing the package and on what is contained in each subdirectory.

It is highly recommended that you obtain a copy of the LAPACK Users' Guide published by SIAM. This Users' Guide gives a detailed description of the philosophy behind LAPACK as well as an explanation of its usage. The LAPACK Users' Guide can be purchased from: SIAM; 3600 University City Science Center; Philadelphia, PA 19104-2688; 215-382-9800, FAX 215-386-7999. It will also be available from booksellers. The Guide costs $15.60 for SIAM members, and $19.50 for non-members. Please specify order code OT31 when ordering. To order by email, send email to service@siam.org.

LAPACK has been thoroughly tested, on many different types of computers. The LAPACK project supports the package in the sense that reports of errors or poor performance will gain immediate attention from the developers. Such reports, descriptions of interesting applications, and other comments should be sent by electronic mail to lapack@cs.utk.edu.

A list of known problems, bugs, and compiler errors for LAPACK is maintained on netlib. For a copy of this report, send email to netlib@ornl.gov with a message of the form: send release_notes from lapack.

A number of working notes were written during the development of LAPACK and published as LAPACK Working Notes, initially by Argonne National Laboratory and later by the University of Tennessee. Many of these reports have subsequently appeared as journal articles. Most of these working notes are available in postscript form from netlib. To receive a list of available reports, send email to netlib@ornl.gov with a message of the form: send index from lapack/lawns. Otherwise, requests for copies of these working notes can be sent to the following address.

```
LAPACK Project
c/o J.J. Dongarra
Computer Science Department
University of Tennessee
Knoxville, Tennessee 37996-1301
USA
Email: lapack@cs.utk.edu
```

P.4.1 Installing LAPACK

Installing LAPACK on Unix systems

See the LAPACK provided notes. The QUICK_INSTALL guide is included here.

== Quick Reference Guide for the Installation of LAPACK ==

VERSION 1.0 : February 29, 1992 VERSION 1.0a : June 30, 1992 VERSION 1.0b : October 31, 1992 VERSION 1.1 : March 31, 1993

DATE: March 31, 1993

This Quick Reference Guide to the installation of LAPACK has been extracted from the Installation Guide contained in LAPACK/INSTALL. It is only intended as a quick reference. The Installation Guide should be consulted for full details.

To install, test, and time LAPACK:

1. Read the tape or uncompress and tar the file.

tar xvf /dev/rst0 (cartridge tape), or

tar xvf /dev/rmt8 (9-track tape)

uncompress lapack.tar.z (from a file), and

tar xvf lapack.tar (from a file)

2. Test and Install the Machine-Dependent Routines (WARNING: You may need to supply a correct version of second.f and dsecnd.f for your machine)

cd LAPACK/INSTALL make testlsame testslamch testdlamch testsecond testdsecnd

3. Create the BLAS Library, if necessary (NOTE: For best performance, it is recommended you use the manufacturers' BLAS)

cp LAPACK/INSTALL/lsame.f LAPACK/BLAS/SRC/ cd LAPACK/BLAS/SRC make

4. Run the Level 2 and 3 BLAS Test Programs

cd LAPACK/BLAS/TESTING make -f makeblat2 cd LAPACK/BLAS xblat2s < sblat2.in xblat2d < dblat2.in xblat2c < cblat2.in xblat2z < zblat2.in cd LAPACK/BLAS/TESTING make -f makeblat3 cd LAPACK/BLAS xblat3s < sblat3.in xblat3d < dblat3.in xblat3c < cblat3.in xblat3z < zblat3.in

5. Create the LAPACK Library

cp LAPACK/INSTALL/lsame.f LAPACK/SRC/ cp LAPACK/INSTALL/slamch.f LAPACK/SRC/ cp LAPACK/INSTALL/dlamch.f LAPACK/SRC/ cp LAPACK/INSTALL/second.f LAPACK/SRC/ cp LAPACK/INSTALL/dsecnd.f LAPACK/SRC/ cd LAPACK/SRC make

6. Create the Library of Test Matrix Generators

cd LAPACK/TESTING/MATGEN make

7. Run the LAPACK Test Programs

cd LAPACK/TESTING make

8. Run the LAPACK Timing Programs

cd LAPACK/TIMING make xlintims < sblasa.in > sblasa.out xlintims < sblasb.in > sblasb.out xlintims < sblasc.in > sblasc.out

repeat timing of blas for c, d, and z

Installing LAPACK on VMS systems

The following steps can be used to install LAPACK on VMS systems.

1. Create directories [.LAPACK.SRC] and [.LAPACK.BLAS.SRC]
2. ftp the Fortran files from the unpacked tar file into the appropriate directory
3. Compile the Fortran files
4. Append the object files into CERN:[NEW.OBJ]LAPACK.OBJ

The Fortran files can be compiled with a simple command procedure such as the one shown below.

Compiling the LAPACK source

```
$ loop:
$ a = f$search("*.F")
$ if a .eqs. "" then exit
$ write sys$output "Compiling ''a'"
$ fortran 'a'
$ goto loop
```

Installing LAPACK on VM/CMS systems

LAPACK is installed as two separate text files:

LABLASRC	Basic Linear Algebra component (BLAS)
LAPACK	LAPACK itself

These correspond to the files found in LAPACK/BLAS/SRC and LAPACK/SRC in the LAPACK tar file respectively.
Having unpacked the tar file, LAPACK can be installed on VM systems using the following steps:

1. Transfer the source files to VM, e.g. using ftp.
2. Edit the source files so that they will compile.
   - Change occurances of **double complex** to **complex*16**.
   - Change **double complex** functions to **COMPLEX function*16**.
   - Change calls to the **DBLE** intrinsic funtion. When the argument is of type **COMPLEX**, change to **REAL(argument)**. e.g. change DBLEx to DBLE(REAL(x)).
   - Change calls to the **INT** intrinsic funtion. When the argument is of type **COMPLEX**, change to **REAL(argument)**. e.g. change INTx to INT(REAL(x)).
   - Change DCMPLX(1) to DCMPLX(1,0.)

The above changes, with the exception of the first two which can be simply performed using an editor, can be done using **FLOPPY** [1].

Converting LAPACK for VSFORTRAN compatibility

```
AXCLIB? SET COMMAND DISK$USER1:[JAMIE.FLOPPY]FLOPPY

AXCLIB? FLOPPY/NOCHECK/TIDY/INDENT=3 LAPACK.FORTRAN ! output to LAPACK.FLOPFOR
```

Install LAPACK on Windows/NT systems

Install LAPACK on MSDOS systems

P.5 MPA

MPA is a package for multiple precision floating point operations. It is a commercial product that can only be distributed in object form. It is automatically included in versions of MATHLIB obtained from asis.
If you intend to rebuilt MATHLIB from scratch, you should copy the object file from CERN.

VM/CMS	MPA TEXT on CERNLIB 322
VAX/VMS	Copy VXCERN::CERNVAX:[PRO.OBJ]MPA.OBJ to CERN:[NEW.OBJ]MPA.OBJ
AXP/VMS	Copy VXCERN::CERNAXP:[PRO.OBJ]MPA.OBJ to CERN:[NEW.OBJ]MPA.OBJ
Unix	Copy

P.6 NAG

[1] Thanks to Julian Bunn for making the necessary changes to a private version of FLOPPY for this purpose.

Bibliography

[1] Various. *CERNLIB - Short Writeups*, Program Library. CERN, 1994.

[2] CN/ASD Group. *HBOOK Users Guide (Version 4.21)*, Program Library Y250. CERN, January 1994.

[3] CN/ASD Group. *KUIP – Kit for a User Interface Package*, Program library I202. CERN, January 1994.

[4] CN/ASD Group and J. Zoll/ECP. *ZEBRA Users Guide*, Program Library Q100. CERN, 1993.

[5] H. J. Klein and J. Zoll. *PATCHY Reference Manual*, Program Library L400. CERN, 1988.

[6] H. Grote and M. Metcalf. *PATCHY for beginners*, Program Library L400. CERN, 1981.

[7] CERN. *CSPACK – Client Server Computing Package*, Program Library Q124, 1991.

[8] H. Grote and I. Mclaren. *EPIO – Experimental Physics Input Output Package*.

[9] J. D. Shiers. *FATMEN - Distributed File and Tape Management*, Program Library Q123. CERN, 1992.

[10] R. Brun, R. Hagelberg, M. Hansroul, I. Ivanchenko, G. Misuri, and J. Vorbrueggen. *FFREAD – Format Free Input Processing*.

[11] J. Shiers et al. *HEPDB - High Energy Physics Data Base (Version 0.03)*. CERN, 1992.

[12] R. Matthews. *KAPACK – Random Access I/O Using Keywords*.

[13] CN/ASD Group. *MINUIT – Users Guide*, Program Library D506. CERN, 1993.

[14] R. Brun, F. Carena, H. Grote, M. Hansroul, J.C. Lassalle, and W. Wojcik. *ZBOOK – User Guide and Reference Manual* Program Library Q210. CERN, 1984.

[15] CN/ASD Group. *PAW users guide*, Program Library Q121. CERN, October 1993.

[16] J. D. Shiers. *VAXTAP - VAX/VMS Tape Handling Package*, Program Library Z312. CERN, 1992.

Index