Replacement Pages for the
Sun386i Developer's Guide

November 1988
These replacement pages are for the *Sun386i Developer's Guide*, part number 814-1009-10, Revision A, May 1988. They are the same pages that were issued in June 1988, when the Sun386i workstation was first released.

The *Sun386i Developer's Guide* is included in the *Sun386i Developer's Toolkit Documentation Set*. Please remove existing pages in the *Sun386i Developer's Guide* and replace them with the pages in this package, according to the table shown below.

<table>
<thead>
<tr>
<th>Replace Pages</th>
<th>With Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>ix–xii</td>
<td>ix–xii</td>
</tr>
<tr>
<td>67–74</td>
<td>67–74</td>
</tr>
<tr>
<td>95–98</td>
<td>95–98C</td>
</tr>
<tr>
<td>111–116</td>
<td>111–116</td>
</tr>
<tr>
<td>137–146</td>
<td>137–146A</td>
</tr>
<tr>
<td>153–154</td>
<td>153–154</td>
</tr>
<tr>
<td>179–180</td>
<td>179–180</td>
</tr>
<tr>
<td>193–207</td>
<td>193–208</td>
</tr>
<tr>
<td>273–287</td>
<td>273–286</td>
</tr>
</tbody>
</table>
Chapter 9  Applications Delivery ....................................................... 137

9.1.  System Software Overview....................................................... 139
9.2.  Application SunOS ............................................................... 140
    Hardware Diagnostics .......................................................... 140
    Core System ....................................................................... 140
    Optional Clusters ............................................................... 141
    Recovery Software ............................................................. 142
9.3.  SunOS Developer's Toolkit ................................................... 143
9.4.  Loading and Unloading Clusters .......................................... 143
9.5.  Releasing Your Software ..................................................... 144
    Installation Script .............................................................. 144
    Start-Up Script ................................................................. 145
    Making the Distribution ..................................................... 146A

Chapter 10  Internationalizing Applications ................................. 147

10.1.  Internationalization Support ............................................... 149
    8-Bit Characters ................................................................. 149
    Alternative Code Sets ........................................................ 149
    Keyboard Support ............................................................. 150
    Native-Language Messages ............................................... 154
10.2.  Application Guidelines ....................................................... 155
    8-Bit Characters ................................................................. 155
    Date and Time Formats ...................................................... 155
    Numeric Formats .............................................................. 155
    Currency Symbols ............................................................ 155
    Text Messages ................................................................... 155

Appendix A  Sun386i System Description .......................................... 157

A.1.  Product Features ................................................................. 159
    System Unit ....................................................................... 159
    Expansion Unit ................................................................. 160
    AT Bus ............................................................................... 160
    Monitors ........................................................................... 161
B.2. Symbols and Expressions................................................................. 172
Values................................................................. 172
Symbols................................................................. 173
Expressions................................................................. 173
B.3. Pseudo Operations............................................................... 175
General Pseudo Operations.......................................................... 175
\texttt{adb} Pseudo Operations.................................................... 177
dbx Pseudo Operations.............................................................. 178
B.4. Machine Instructions............................................................ 178
Differences between the SunOS and Intel 80386 Assemblers............. 178
Operands..................................................................................... 178
Introduction to Instruction Descriptions...................................... 180
Processor Extension Instructions.................................................. 182
Segment Register Instructions..................................................... 184
I/O Instructions.............................................................................. 184
Flag Instructions.............................................................................. 184
Arithmetic/Logical Instructions..................................................... 185
Multiply and Divide Instructions................................................... 186
Conversion Instructions.................................................................. 186
Decimal Arithmetic Instructions................................................... 186
Coprocessor Instructions............................................................... 186
String Instructions........................................................................... 186
Procedure Call and Return Instructions......................................... 187
Jump Instructions............................................................................. 187
Interrupt Instructions........................................................................ 187
Protection Model Instructions....................................................... 187
Miscellaneous Instructions........................................................... 188
B.5. Translation Tables for SunOS to Intel Float Mnemonics.............. 188
Real Transfers................................................................................. 188
Integer Transfers............................................................................. 189
Packed Decimal Transfers............................................................. 189
Addition......................................................................................... 189
Subtraction....................................................................................... 189
Multiplication.................................................................................. 189
Division.......................................................................................... 189
Other Arithmetic Operations .......................................................... 190
Comparison Instructions ................................................................. 190
Transcendental Instructions ............................................................. 190
Constant Instructions ..................................................................... 190
Processor Control Instructions ......................................................... 191

Appendix C File System Layout...................................................... 193

C.1. Terms .................................................................................. 195
C.2. Layout Overview .................................................................. 196
    System Disk ...................................................................... 196
    Additional Disks .................................................................. 197
C.3. / File System ........................................................................ 197
C.4. /usr File System ................................................................. 200
C.5. /files<n> File System ............................................................ 203
C.6. /export Directory ................................................................. 204
C.7. /vol Directory ...................................................................... 206
C.8. Application Directory Structure ............................................. 207

Appendix D Common Object File Format (COFF) ....................... 209

D.1. COFF Features .................................................................... 211
    COFF Structure ................................................................. 212
D.2. Terms and Conventions .......................................................... 212
D.3. File Header .......................................................................... 213
    Magic Numbers .................................................................. 213
    Flags ................................................................................ 214
    File Header Declaration .................................................... 214
D.4. Optional Header Information .................................................. 214
    Standard SunOS System a.out Header .................................. 214
    Optional Header Declaration ............................................. 215
D.5. Section Headers .................................................................... 216
    Flags ................................................................................ 216
    Section Header Declaration .............................................. 217
    .bss Section Header .......................................................... 218
D.6. Sections .............................................................................. 218
As shown above, the Sun386i system supports a version of the ANSI Computer Graphics Interface (CGI) standard graphics package for generation of two-dimensional images, called SunCGI. In addition, the Sun386i system supports SunGKS, the well-established Graphical Kernel System (GKS) standard for interactive two-dimensional graphics. Both SunCGI and SunGKS are unbundled products.

The next section discusses the Sun Organizer™, one of the window-based applications shown above, which you can use to create icons for your application’s files.
The organizer Program

Organizer (organizer(1)) displays the contents of directories by using icons to denote file types. Pop-up menus, panel buttons, and property sheets provide SunOS file system commands such as mv(1), cp(1), rm(1), lpr(1), edit(1), open(2), rename(2), chmod(2), find(1), and mkdir(1).

Users can display directories with or without icons, and they can sort the contents of these directories by name, file type, size, or date. In addition, with the Show Map feature users can graphically view or browse the file system hierarchy in one window. You can create icons for your application's files and include them in the .orgrc file, described in the following section. When you do, organizer automatically displays your file-type icons in its Show List and Show Map windows.


Displaying File Types for Your Applications

organizer(1) can display three-color icons representing your application's files if you:

- Create icons for your file types.
- Load your .orgrc file as share/data/.orgrc and your icon files as share/icons/icon_names.icon as part of the installation procedure. The administrator loading your application must create an application_name directory and go (cd(1)) to that directory before installing your files, as shown on page 96. Page 207 describes the suggested hierarchy for your application files.
- As part of your installation notes, instruct system administrators to create a volume for your application and to append your .orgrc file to -users/defaults/.orgrc and to -groupname/defaults/.orgrc for each group in use. Then users must issue the cat(1) command to append your .orgrc file to their individual version of the file in ~/.orgrc after your application is installed. Finally, users must quit from and re-enter Organizer to see your application's icons. (source(1) does not work on .orgrc.) See pages 96-98B for details.

After users append your .orgrc to their own copy of the file, they can open a file or run an application by double-clicking on the icons you supply. If an administrator appends your .orgrc file to -users/defaults/.orgrc, the default version of the file, then user accounts created after your application is installed will automatically receive a copy of .orgrc that contains your icon information.

The steps to create and incorporate a file-type icon into organizer are:

1. Determine the name-matching expression to identify your file type. For example, the defaults that come with the system are *.c for C program files, *.h for header files, *.o for object files, and *.icon for icon files. It is the responsibility of the system administrator to rename duplicate file-type extensions that might occur.
2. Create one set of icons per file type that you are defining with the iconedit(1) utility. To enable three-color icons, you must create:
   - A background icon for the icon’s background and
   - A foreground icon, which goes on top of the background icon
   To create two-color icons, you need only specify a background or foreground icon.
3. Quit from the organizer if it is running.
4. Add entries describing your icons to the version of .orgrc in your home directory (~/.orgrc). The format of .orgrc is shown in the following sections.

5. Re-enter organizer to view the icons created or changed.

After restarting organizer, you can use coloredit(1) to change the colors of icons. When you quit from organizer, the changes made are saved to the version of .orgrc in your home directory.

Once you determine the colors you like best, make a copy of your home .orgrc file for distribution.

.orgrc Parameters

.orgrc files contain two kinds of parameters:

- File-type parameters — You must create one set of file-type parameters for each icon accompanying your software. These parameters are described in the next section, with required parameters shown in bold.
- Color-palette parameters — These parameters define the default colors that the Sun386i system uses for directory, text, executable, and device files. You can use the colors supplied by these parameters for your icons, add to this file if your application uses files other than the four default types, or ignore this section of .orgrc and use your own RGB values in your file-type parameters. These parameters are described on the next page.

All .orgrc parameters, as well as all values for parameters, are case insensitive.

File-Type Parameters

Begin File Type Definition

These four words are required to denote the start of each file type within .orgrc.

Name

The expression used for name matching; you can use any valid SunOS wildcard. For instance, for a file type ending with .pbj, you could enter *.pbj as a value for this parameter. Another example is *.s for SCCS (Source Code Control System) files. Alternatively, you could enter the exact file name, as with the core file type provided with the Sun386i system.

Background Icon

The path name of the background icon that organizer will display.

Foreground Icon

The path name of the foreground icon (placed on top of the background icon) that organizer will display.

Name Color

The RGB values for the color of the text of the file name as it appears on the icon. Also used to color the rectangle that surrounds a file when it is selected.
Icon Background Color
The RGB values for the background color of the icon.

Icon Foreground Color
The RGB values for the foreground color of the icon.

Highlight Name Color
The RGB values for the color of the text of the file name when the file or directory is selected (highlighted).

Execute Application
The name of the application to call to open or execute this file type. Some file types that users should not open are object files, intermediate database format files, libraries, and binary data files.

If the application that you're opening accepts file name arguments, you can use the $(FILE) keyword to tell organizer to open that application's files when users double-click on a file name. Just use $(FILE) instead of the file name in the command line, as shown in the following example:

```
Execute Application = textedit $(FILE)
```

Edit Application
The name of the application to call to edit this file type. If you do not supply a value for this parameter, users will be unable to edit any files of this file type from within organizer. As with Execute Application, you can specify the $(FILE) keyword so that users can edit the selected file.

Print Application
The name of the application to call to print this file type. If you do not supply a value for this parameter, users will be unable to print any files of this file type from within organizer. If the application can print files automatically, you can include the $(FILE) keyword to enable users to automatically print the application's files through organizer.

The Name Color, Icon Foreground Color, Icon Background Color, and Highlight Name Color parameters are optional because you can use coloredit to determine icon colors; when you quit from organizer, it automatically writes the colors chosen to .orgrc in your home directory. You can then include the RGB values from your home .orgrc in the version that you supply with your application.

End File Type Definition
These four words are required to denote the end of the definition.

Instead of providing numerical values for your icon colors in the preceding color-related parameters, you could use the names of the color palette parameters shown
below for directory, text, executable, and device file icons. You might want to use these parameter names instead of RGB values in your file-type definitions because:

- It's easier to use them than to use trial and error to determine colors for your applications.
- The colors of your directory, text, executable, and device icons will match the default values for SunView versions of these file types; the graphic of your icons rather than their colors will differentiate them.
- If users change the color palette RGB values, your icons will match the colors they choose.
- You might not want to use any more of the colormap on your icons, particularly if your application uses a lot of color (the Sun386i system permits a maximum of 256 colors at one time).

The default color palette portion of .orgrc supplied with the Sun386i system follows.

```
Begin Color Palette
  Background Color = 255, 255, 255
  Directory Name Color = 0, 146, 236
  Directory Icon Foreground Color = 255, 227, 185
  Directory Icon Background Color = 114, 45, 0
  Directory Highlight Name Color = 255, 247, 9
  Text Name Color = 0, 166, 143
  Text Icon Foreground Color = 255, 255, 255
  Text Icon Background Color = 0, 0, 0
  Text Highlight Name Color = 255, 255, 0
  Executable Name Color = 255, 0, 104
  Executable Icon Foreground Color = 243, 255, 254
  Executable Icon Background Color = 157, 162, 187
  Executable Highlight Name Color = 255, 247, 9
  Device Name Color = 111, 111, 111
  Device Icon Foreground Color = 243, 255, 254
  Device Icon Background Color = 157, 162, 187
  Device Highlight Name Color = 255, 255, 0
  Scrollbar Color = 0, 87, 185
End Color Palette
```

Most of the color palette parameters correspond to file-type parameters. For instance, to use the default colors for a text file icon, include the lines below in your file-type definition:

```
Name Color = Text Name Color
Icon Foreground Color = Text Icon Foreground Color
Icon Background Color = Text Icon Background Color
Highlight Name Color = Text Highlight Name Color
```
The first and last color palette parameters, Background Color and Scrollbar Color, do not have file-type counterparts.

If your application uses files that don't fit one of the four default categories, you can add to the color palette section. However, do not change the defaults that are there, since these are the colors that Sun wants users to see; users can modify these colors if they so choose.

Alternatively, you might want to provide your own icon colors to make them stand out (but be aware that users can change these also). If so, provide RGB values instead of color palette parameter names in your file-type definitions.

Sample .orgrc Entries

This section provides sample entries for the .orgrc file for an application that will be available over a network. If your application will be available only on the system where it is installed, then use the path /usr/local/application_name instead.

```plaintext
Begin File Type Definition
Name = *.c
Background Icon = /vol/application_name/
    share/icons/bkgd.c.icon
Foreground Icon = /vol/application_name/
    share/icons/frgd.c.icon
Name Color = 236, 121, 85
Icon Background Color = 205, 205, 254
Icon Foreground Color = 51, 19, 92
Highlight Name Color = 255, 247, 9
Execute Application = cmdtool vi $(FILE)
Edit Application = cmdtool vi $(FILE)
Print Application = lpr $(FILE)
End File Type Definition

Begin File Type Definition
Name = *.icon
Background Icon = /vol/application_name/
    share/icons/bkgd.icon
Foreground Icon = /vol/application_name/
    share/icons/frgd.icon
Name Color = 124, 61, 140
Icon Background Color = 243, 255, 254
Icon Foreground Color = 157, 162, 187
Highlight Name Color = 255, 254, 8
Execute Application = iconedit $(FILE)
Edit Application = iconedit $(FILE)
End File Type Definition

Begin File Type Definition
Name = *.bits
Revision B, June 1988
```
In addition to man pages, the Sun386i system offers three areas of on-screen messages:

- Detailed instructions on how to create a new user account and how to log in
- Improved kernel error messages
- Cursor-sensitive help for SunView applications

The log-in screens are part of the new administration features, and are described more fully on page 99. This section describes kernel error messages and cursor-sensitive help for applications, including how to add to or change both types of these messages.

About 40 of the most frequently displayed kernel error messages have been reworded for the Sun386i system. The original messages are intercepted by a message daemon and used as keys into a message text file. The keyed messages, rewritten for clarity, then are delivered to a log file, to a SunView window, or to both.

There are several possible reasons for changing error message output:

- To translate the messages that Sun has reworded into another language (refer to the Native-Language Messages section on page 154 for more information)
- To intercept and reword the output of additional error messages generated by the kernel (those sent to /dev/klog by the syslogd(8) daemon)
- To intercept and reword the output of error messages generated by local programs, provided the program sends its messages to /dev/log via syslog(3)
- To intercept and reword the output of messages placed in the internet socket by programs on other systems

The steps below apply to the addition of substitute messages (the latter three cases).

1. Look through kernel source code (if you have the appropriate license), or the source code for the program containing messages that you want to replace. You can only substitute messages for programs that send their messages to /dev/log via syslog.
2. Add the text of the existing, unexpanded messages that you want to replace to /etc/In, preceding each message with a unique number. (You also can include suppression instructions for these messages, as described later in this section.)

3. Add the substitute text for these messages to /etc/Out, preceding each message with the number corresponding to the original message in /etc/In.

4. Use the kill -1 command on the syslogd process to activate the changes made.

You don’t have to use the default files /etc/In and /etc/Out for original and substitute text, respectively, but it’s easier if you do. If you use other files, you must include the file names in /etc/syslog.conf. This file tells the system log daemon, syslogd(8), the information needed to perform the translation or suppression. syslogd intercepts error messages, checks /etc/In and the file you specify in /etc/syslog.conf (if any) for the text of those messages, and either suppresses, substitutes, or displays the original messages.

If you use files other than /etc/In and /etc/Out, you must add an entry to /etc/syslog.conf that contains the following five fields, separated by tabs:

```
translate source facility input_file output_file
```

*translate* — Denotes a translation entry.

*source* — Specifies the source(s) of an error message, separated by commas; recognized sources are:

- klog, indicating a kernel message sent to /dev/klog
- log, indicating a message generated by a local program and sent to /dev/log
- syslog, indicating a message placed in the internet socket by programs on other systems
- *, indicating all three of the above sources

*facility* — Specifies messages generated by other system facilities, separated by commas:

- user, indicating messages from user processes. This is the default for messages from programs or facilities not listed in syslog.conf.
- kern, indicating messages from the kernel
- mail, indicating messages from the mail system
- daemon, indicating messages from system daemons such as ftpd(8) and routed(8)
- auth, indicating messages from the authorization system (login(1), getty(8), and su(1))
- lpr, indicating messages from the line printer spooling system
- cron, indicating messages from the cron/at facility
- mark, indicating messages produced internally by syslogd
- *, indicating all facilities except mark

Revision B, June 1988
If you are using Frame, you can work on handbook files in the default help directory. If they have the appropriate links and the handbook contents page is in the Top Level (also in the default help directory), you can look at them at any time in the Help Viewer.

If you are using Interleaf, your working files will be in your desktop directory; you will have to transfer them to the default help directory before displaying them in the Help Viewer. However, first you must convert them to Printerleaf format by performing the steps below.

1. Pop up the Printer menu.
2. Pull right on Printerleaf and select Document. This saves the file in Printerleaf format and appends the .pl extension to its name.
3. When you move, copy, or rename the file in the default help directory, strip off the .pl.

Checking Topic Appearance and Function

As you create topics you will likely want to check their appearance and the functioning of hypertext links in the Help Viewer. To do this:

1. Copy your handbook files into the default help directory that you've specified with Defaults (if they are not already there).
2. Add your handbook to the copy of the Top Level file in the default help directory so that you can link to it from the viewer's Top Level table of contents.

The following section discusses the second step.

Adding Handbooks to the Top Level

No one can access your handbook from the Help Viewer until you insert the handbook title in the Help Viewer's Top Level table of contents. Top Level is a special text file (not a Frame or Interleaf file) in a subdirectory of the default help directory that you can edit for this purpose. Top Level is initially in /vol/help/language/USA-English.

The file lists the standard set of Sun386i handbooks as a series of entries. Each entry consists of a text string in single quotes on the left and a path name and page number in square brackets on the right. The strings are displayed as underlined contents items; the path names are links to the files displayed after following the link.

You edit Top Level as you would any other text file. The only things to note are the use of:

1. Backslash characters in front of apostrophes, or backslashes that are part of your handbook's title
2. Number signs in front of comments

The next section describes installing your help files, and includes suggestions for appending your handbook entry to Top Level via a shell script.

Installing Your Help Files

This section contains the required steps that an installation script must perform to load help for your applications, as well as the steps that installation instructions must include to ensure that users can access your help.

Revision B, June 1988
Adding to the Top Level During Installation

The Top Level might already be customized to a certain extent. Therefore, it's suggested that you write a shell script that appends your handbook entry to the Top Level file in /vol/help.master (a writable version of /vol/help) when a customer installs your application. Such a script would resemble the following:

```bash
echo "'application_name' Handbook" \ 
     [application/application_name_Handbook]" >> \ 
     /vol/help.master/language/USA-English/Top_Level
```

Or, to check for the presence of your handbook before adding it:

```bash
cd /vol/help.master/language/USA-English
if grep 'application_name' Handbook' Top_Level > \ 
    /dev/null 2>&1
then
    echo "'application_name' Handbook' is already \ 
    installed in 'Top_Level'
else
    echo "'application_name' Handbook' \ 
    [application/application_name_Handbook]" >> \ 
    /vol/help.master/language/USA-English/Top_Level
echo "'application_name' Handbook' installed in \ 
    'Top_Level'
fi
```

Required Installation Steps for Applications with Help Files and Organizer Icons

As described on page 80, the default help directory is /vol/help. It is recommended that administrators retain this as the default directory for easier maintenance, and for disk space conservation and network-wide availability of Sun386i and third-party help files. For the Help system to access your files through /vol/help, you must add a few steps to your installation script, and must instruct system administrators to perform certain steps before and after installing your software. These steps are also mandatory if your application includes Organizer icons for your application's files.

Pre-Installation Instructions for System Administrators

Be sure that your installation instructions tell an administrator to do these steps before installing your application.

1. Log in as superuser on the system where you'll be installing the application.
2. Go (cd(1)) to the /files<n>/vol directory if the application will be available over the network, or to the /usr/local directory if it will be available only on this system. Choose the /files directory (/files or /files1, /files2, and so on) based on space and use.
3. Enter mkdir application_name to create an application_name subdirectory.
4. Enter cd application_name.
5. If the application is on diskette, enter:
   ```bash
   bar xvf /dev/rfd0a installation_script
   ```
   If the application is on high-density tape, enter:
   ```bash
tar xvf /dev/rst8 installation_script
   ```
If the application is on low-density tape, enter:
```bash
tar xvf /dev/rst0 installation_script
```

6. Enter `installation_script`.

The final step will invoke the installation script that you supply, which must perform the steps listed in the next section.

### Installation Script Steps

As part of the installation procedure, your script must load the help files and the rest of the application as follows:

For diskettes, use the command:
```bash
bar xvf /dev/rfd0a directories
```

For high-density tapes, use the command:
```bash
tar xvf /dev/rst8 directories
```

For low-density tapes, use the command:
```bash
tar xvf /dev/rst0 directories
```

where `directories` is `language/language/help/application_name.info` for Spot Help files and `language/language/help/handbook/handbook_files` for handbook files. Pages 207-208 describe the suggested directory structure for applications. This structure is required for applications that include on-screen help files. Note that your script could use additional options with `bar` and `tar`, depending on the options you use when putting your files on tape or diskette.

You also must include either the start-up script supplied with the Sun386i system, `/usr/etc/start_applic`, or one that you provide. Pages 144-145 provide additional information about start-up and installation scripts.

### Installation Instructions for System Administrators

If your application includes on-screen help files, or if you're adding Organizer icons for your application's files, whoever loads your application must perform the steps in the rest of this section to access your help or icons. Although these instructions also appear in *Sun386i Advanced Administration*, you should include them in the written installation instructions that you provide with your application.

This section assumes that the application will be available throughout a Sun386i network or for a domain specified. If for licensing reasons your application should only be available on the system where it is installed, the administrator should install the application in `/usr/local` (instead of `/files<n>/vol`) and should not export or create a volume for the application. Instead, administrators should only perform the steps starting with the Registering an Application section on page 98A.

If administrators are installing your application on a non-Sun386i network (an existing YP network with a master server that is not a Sun386i system), they must manually mount the application on all systems that will use it, and must devise their own registration procedure.

### Exporting the Application

Exporting is controlled through the `/etc/exports` file (see the `exports(5)` man page for details). The following steps show how to create a link to an application that's being exported, export the application by editing the `/etc/exports` file.
file, and issue the exportfs(8) command to activate modifications made to the
/etc/exports file. If you want to include these steps in your installation script
rather than have a system administrator perform them, see the Installation Script
section on page 144.

Before performing these steps, an administrator must make sure that no one is
adding a diskless client to the system and that no one is creating a user account on it
through New User Accounts or SNAP. This is to avoid possible corruption of the
/etc/exports file, used by both procedures and modified in the steps below.

1. Log in as superuser to the system where the application files reside.
2. Ensure that /export/vol exists by entering the command:
   
   ```
   mkdir /export/vol > /dev/null 2>&1
   ```
3. Create a symbolic link using the command format:
   
   ```
   ln -s /files<n>/vol/application_name /export/vol/application_name
   ```
4. Enter the following line to add an entry for the application in the
   /etc/exports file:
   
   ```
   # echo "/export/vol/application_name -ro,access=domain" >> /etc/exports
   ```
5. Enter the following command to export the application:
   
   ```
   exportfs /export/vol/application_name
   ```

The application is now available to users on any system within the domain net­
group (see the netgroup(5) man page) via the mount(1) command. The steps in
the next section make the application accessible from an automatically mounted vol­
ume.

Creating a Volume

Creating a volume for an application’s files makes administration and maintenance of
that application easier. Volumes are attached to the Sun386i file system by the auto­
mounter (automount(8)), so there’s no need to edit /etc/fstab files on client
machines when a volume is moved. Administrators must export the application, as
described in the previous section, before performing these steps.

1. Enter rlogin 'ypwhich -m auto.vol' to log in to the Yellow
   Pages master.
2. Become superuser by entering su and the superuser password.
3. Create a volume for the application by entering the following command,
   which adds an entry to the /etc/auto.vol file:
   
   ```
   # echo "application_name -ro 
   system:/export/vol/application_name" >> /etc/auto.vol
   ```
   
   where system is the name of the system where the application files reside.
4. Rebuild the /etc/auto.vol map by issuing the commands:
   
   ```
   # cd /var/yp; make
   ```

The next section describes how administrators will use the start-up script that you
supply to register your application. If your application includes on-screen help files
or Organizer icons, you must supply a start-up script and an administrator must reg-
ister your application as shown in the next section. You are strongly encouraged to
include a start-up script even if you are not providing help files or icons for your
application. Page 145 provides more information about start-up scripts.

Registering an Application

Registering an application enables any user that can access the application to do so
simply by entering application_name, which is the start-up script that invokes the
application. Users need not know the location of the files nor the architecture of the
machine where the files reside. Another benefit is that users do not have to modify
their own environment (such as .login and .cshrc files) to use new applications.
To register an application, an administrator must put a copy of the start-up script
that you supply with your application in /vol/local.master/bin, which is
part of every user’s $PATH.

By default, the /vol/local.master directory and the master copy of the
/etc/auto.vol file are on the same system, so administrators might be able to
enter the following commands on the same machine where they created the volume
in the previous section. If administrators installed the application in /usr/local,
they should go directly to step 2.

1. Enter the command:

```
# ypmatch local.master auto.vol
```

The system displays:

```
system:/export/vol/local
```

If the system name displayed is the system where you created the volume,
then you can perform these steps on the same system; if the name is differ-
ent, then enter:

```
# rlogin system
# su
```

where system is the name of the system displayed.

2. If you’ve exported and created a volume for the application, enter:

```
# cd /vol/local.master/bin
```

If instead you’ve installed the application in /usr/local, making it
available only on an individual system, you must use the format:

```
# cd /usr/local
```

3. Then issue the cp(1) command to copy the start-up script specified in the
installation instructions to application_name. For instance, if the application
uses the start-up script supplied with the Sun386i system, the com-
mand line would be in the format:

```
# cp /usr/etc/start_applic application_name
```

The next two sections describe the final steps that an administrator must perform
after installing an application with help files or Organizer icons, respectively.
Additional Steps for Applications with Help Files

After an administrator performs the steps in this section, your application and help files will be available to any user on the network, without users having to modify any of their environment variables. You might want to follow these steps yourself after your help files are completed, to be sure that everything is working correctly. If you do, be sure to move your files to /vol/help/language/language (page 208 lists values for language), and use Defaults on the SunView Menu to make sure that the Help directory is /vol/help if you’ve changed it while writing or checking your help text.

1. Log in as superuser to the system that has the master /vol/help.master file. To determine the correct system, use the command:

   `ypmatch -m help.master auto.vol`

2. Create a link file for .info files in the directory /vol/help.master:

   `ln -s filename linkname`

   where filename is

   `/vol/application_name/language/language/help/application_name.info`

   if the application will be available on a network; if the application will be available only on systems where it is installed, filename must be

   `/usr/local/application_name/language/language/help/application_name.info`

   Regardless of where the application is installed, linkname has the format:

   `/vol/help.master/language/language/application_name.info`

3. Create a second link file in /vol/help.master for handbooks. The link has the same format as for .info files but filename is

   `/vol/application_name/language/language/help/application_name/handbook`

   for applications available on a network; for an application available only on systems where it is installed, filename is

   `/usr/local/application_name/language/language/help/application_name/handbook`

   For either case, linkname is

   `/vol/help.master/language/language/application_name/handbook`

Additional Steps for Applications with Icons

To access your application’s icons, an administrator must append your

`$application_name_ROOT/share/data/.orgrc` file to

~users/defaults/.orgrc (the version shipped with the Sun386i system) and

to ~groupname/defaults/.orgrc for each group in use. Then an administrator
must tell current users (via electronic mail) to:

1. Use the cat(1) command to append your .orgrc file to their personal
   copy of the file in ~/.orgrc.

2. Quit from and re-enter Organizer to see your application’s icons.
   (source(1) does not work on .orgrc.)

When an administrator appends your .orgrc file to ~users/defaults/.orgrc
and ~groupname/defaults/.orgrc, users subsequently added to the system do not have to perform the above two steps.
6.3 Administration Facilities

The ease-of-use administration facilities described in this section are:

- **snap(1)**
- Automatic System Installation
- New User Accounts

All of these facilities are built on top of two new SunOS 4.0 features: secure RPCs (Remote Procedure Calls) and the Yellow Pages (YP) updater program. Secure RPCs use a public key encryption technology that provides user authentication in the network. The YP updater provides a way to update YP maps from programs. To use these features, your network must be running the Yellow Pages and a Sun386i system must be the YP master. *Sun386i SNAP Administration* describes RPCs and the Yellow Pages in detail.

The next three sections present an overview of the administration facilities on the Sun386i system. For details, see *Sun386i SNAP Administration*.

**The snap Program**

*snap(1)* provides a window interface for users with the required privileges to browse, add, delete, and modify user accounts, user groups, systems, modems, terminals, and printers. When a user confirms changes made, *snap* makes the appropriate changes to the Yellow Pages facility and directories. *snap* privileges are implemented through membership in special groups. As shipped, *snap* gives all users all *snap* privileges; most sites probably will want to restrict privileges to some degree.

*snap* also enables file backup and restore. The backup facility provides easy-to-use personal and system-wide backup functions. Both full and incremental backups are possible. Users can also restore files with *snap*.

**Automatic System Installation**

Using Automatic System Installation, a user can add a system to an existing Sun386i network in about 30 minutes after unpacking. The first time a user powers on the Sun386i system, one of the following four scenarios occurs, depending on the machine’s configuration.

For a standalone, the system asks for confirmation that it is not going to be added to a network, and then continues with the boot procedure.
Memory

The Sun386i system allocates 640 Kbytes of its virtual memory for each application running in DOS Windows. In addition, up to 8 Mbytes of expanded memory is available for each application that uses the Lotus®-Intel-Microsoft (LIM) expanded memory specification. Since the Sun386i system allocates virtual memory, no added hardware is required for this support. If you want to take advantage of expanded memory, you must design applications to access LIM memory by switching parts of the program in and out of LIM address space, which starts at the standard location D00000. LIM memory is available to each DOS Windows application that can use it, even if several windows are running simultaneously.

Naming Your PC Applications

Because MS-DOS and SunOS systems have different file-naming conventions, the SunOS system provides a file name mapping scheme that enables you to specify programs from within either operating system. However, mapped file names are only temporary references; name mapping does not produce the same result each time. Therefore, do not build mapped names into your applications. The best course of action is to follow MS-DOS file naming conventions whenever possible. If you follow the suggestions below, file mapping will not be an issue for you or the users of your applications.

- Names can be up to eight characters (without an extension), or up to eleven characters (with a period and three-character extension). Only programs with .EXE, .COM, or .BAT extensions are executable from within MS-DOS.
- MS-DOS names are not case-sensitive, but almost all SunOS commands are lowercase; therefore use lowercase letters for all of your file names.
- Do not use these characters in file names: ", / ] : I < > + = ; . You can use a period only as a separator between the file name and an extension.

Issuing SunOS Commands from DOS Windows

The system comes with a number of SunOS commands that you can issue from within DOS Windows, provided that /etc/dos/unix is part of your MS-DOS path. These commands are MS-DOS .COM programs that point to the actual SunOS commands. They accept the ampersand (&), so you can run them in the background. The preinstalled commands are listed below.

<table>
<thead>
<tr>
<th>Preinstalled SunOS Commands</th>
</tr>
</thead>
<tbody>
<tr>
<td>at  date  lprm  pr  tar</td>
</tr>
<tr>
<td>awk  diff  ls  ps  tee</td>
</tr>
<tr>
<td>calendar  echo  mail  pwd  time</td>
</tr>
<tr>
<td>cat  egrep  mailtool  rlogin  tr</td>
</tr>
<tr>
<td>chgrp  fgrep  make  rm  umount</td>
</tr>
<tr>
<td>chmod  file  man  rmdir  unix</td>
</tr>
<tr>
<td>chown  find  mesg  rsh  vi</td>
</tr>
<tr>
<td>cmdtool  grep  mkdir  sed  wc</td>
</tr>
<tr>
<td>cmp  head  more  size  whatis</td>
</tr>
<tr>
<td>comm  kill  mount  sort  whereis</td>
</tr>
<tr>
<td>cp  ln  mv  split  who</td>
</tr>
<tr>
<td>csh  lpq  nice  stty  write</td>
</tr>
<tr>
<td>cut  lpr  passwd  tail  yppasswd</td>
</tr>
</tbody>
</table>

Revision B, June 1988
To install additional SunOS commands for your applications, become superuser by entering the `su` command and then enter:

```
cd /etc/dos/unix
ln -s unix.com newcommand.com
```

at the SunOS prompt.

**Text-Only Applications**

dos(1) opens a window whenever you invoke most PC programs. However, this is not the case for text-only applications delivered with the Sun386i system. Text-only applications are those that do not attempt to address the cursor, clear the screen, or display graphics. `DIR` is a good example of such a program. `vi` is not a text-only application, since it controls the cursor position and makes assumptions about the screen geography.

Text-only applications do not require an 80x25 display. Therefore, if implicit execution is set with the `DOSLOOKUP` environment variable, dos executes text-only applications in a cmdtool or shelltool window, rather than automatically popping open a new DOS window. You can add to the list of text-only applications that dos recognizes by including the application's name to your `setup.pc` files, as a value for `TEXT`. (Refer to page 114 for more information about the `setup.pc` file.) The list of text-only applications that are shipped with the Sun386i system is shown in Table 7-2.

<table>
<thead>
<tr>
<th>ATTRIB</th>
<th>DEBUG</th>
<th>LABEL</th>
<th>SUBST</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASSIGN</td>
<td>DIR</td>
<td>LINK</td>
<td>SYS</td>
</tr>
<tr>
<td>BACKUP</td>
<td>DISKCOMP</td>
<td>MODE</td>
<td>TIME</td>
</tr>
<tr>
<td>BREAK</td>
<td>DISKCOPY</td>
<td>RECOVER</td>
<td>TREE</td>
</tr>
<tr>
<td>CHKDSK</td>
<td>EXE2BIN</td>
<td>REPLACE</td>
<td>TYPE</td>
</tr>
<tr>
<td>COMMAND</td>
<td>FDISK</td>
<td>RESTORE</td>
<td>VER</td>
</tr>
<tr>
<td>COMP</td>
<td>FIND</td>
<td>SELECT</td>
<td>VERIFY</td>
</tr>
<tr>
<td>COPY</td>
<td>FORMAT</td>
<td>SHARE</td>
<td>XCOPY</td>
</tr>
<tr>
<td>DATE</td>
<td>JOIN</td>
<td>SORT</td>
<td>XDIR</td>
</tr>
</tbody>
</table>

**Running `make(1)` on MS-DOS Targets**

In addition, if you specify MS-DOS files as text-only files, you can run compilers, assemblers, and SunOS `make(1)` files on them in cmdtool or shelltool windows. For example,

```
file.exe: file.c

dos -w -c cc file.c
```

where `file.exe` is the MS-DOS target file, and `file.c` is the file that the target file depends upon. The `-w` option to the `dos` command declares `file.c` as a text-only file, and the `-c` option indicates that the command, in this case `cc`, follows.
File Permission Differences

Generally, access to files is the same under SunOS and MS-DOS systems. The exceptions are:

- MS-DOS does not recognize execute restrictions. That is, any user with read permission to a file can execute that file. Without read permission, users cannot execute files.
- Drive C: does not support SunOS file permissions, since the SunOS system cannot directly access files on drive C:. However, because the SunOS system views drive C: as one large file, you can restrict access to all drive C: files to a specific owner or group.

7.3. Peripheral Issues

The MS-DOS drive designations are:

- **Drives A: and B:** — Reserved for diskettes.
- **Drive C:** — A “virtual” hard disk of up to 20 Mbytes, that the SunOS system cannot access; use this drive only to install copy-protected or install-protected PC software.
- **Drives D: through S:** — Virtual hard disks tied to system SunOS directories that can expand as required; use these drives for data files and unprotected PC applications.

All MS-DOS drivers listed in CONFIG.SYS, the MS-DOS configuration file, must actually be on drive C:, where CONFIG.SYS resides. This is because MS-DOS loads drivers before it begins to communicate with the SunOS system (toward the end of the AUTOEXEC.BAT file), and drive C: is the only drive that MS-DOS can access until SunOS communications are activated. The message Bad or missing xxx.sys appears if you try to access a device that has a driver that is not on drive C:.

You can add MS-DOS peripherals either by:

- Adding an AT card that uses an MS-DOS driver provided by the card manufacturer
- Adding an AT card that uses a device-specific driver that you write

Regardless of the method used, you must add information about new drivers to three files—CONFIG.SYS (an MS-DOS file), and setup.pc and boards.pc (two SunOS files described in the following sections). Then you must invoke DOS Windows from the Desktop menu before using the new driver, or enter dos -s to save the new driver in .quickpc, a quick-start file containing a snap-shot image of MS-DOS. (Refer to Sun386i Advanced Skills for more information.)

The setup.pc file contains configuration information on all devices attached to a system that users might want to access via MS-DOS, including the SunOS files associated with those devices. The boards.pc file contains a list of the boards that only MS-DOS, not the SunOS system, can access on the system. MS-DOS cannot access a peripheral listed in the setup.pc file unless it is also in the boards.pc file. The boards.pc file is in the /etc/dos/defaults directory, and setup.pc is in the user’s home directory, ~/pc/setup.pc.
The first time you open DOS Windows, dos creates a pc directory under the home
directory, and places a copy of setup.pc there. You can edit this file, but general­ly
should not delete anything in it. A description of the default setup.pc file
follows, with number signs (#) indicating comment lines. (Descriptions shown here
are not part of the default file.) For more general information about the setup.pc
file, refer to Sun386i Advanced Skills.

<table>
<thead>
<tr>
<th>MS-DOS Device</th>
<th>SunOS Device Path Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>/dev/rfd0c</td>
</tr>
<tr>
<td># Diskette device name.</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>~/pc/C:</td>
</tr>
<tr>
<td># Drive C: file name.</td>
<td></td>
</tr>
<tr>
<td>COM1</td>
<td>/dev/ttya</td>
</tr>
<tr>
<td># Specifies the serial device attached to the serial port.</td>
<td></td>
</tr>
<tr>
<td>LPT1</td>
<td>lpr</td>
</tr>
<tr>
<td># Specifies how to process MS-DOS LPT1 text; the default is the default printer.</td>
<td></td>
</tr>
<tr>
<td>LPT2</td>
<td>cat &gt;&gt;~/lpt-2</td>
</tr>
</tbody>
</table>
| # Specifies how to process MS-DOS LPT2 text; the default is to append to the
| # ~/lpt-2 file in the user's home directory. |
| LPT3          | psfx80 | lpr |
| # Specifies how to process MS-DOS LPT3 text; the default is Epson™ FX-80
| # emulation on the default printer. |
| SAVE          | ~/pc/.quickpc          |
| # Specifies .quickpc, a quick-start file created with the dos -s command
| # that contains a snapshot image of MS-DOS after it has read CONFIG.SYS and
| # most of AUTOEXEC.BAT (up to the RUNDOS line). This file is used when any
| # dos command other than dos -b (the default command issued when started
| # from the Desktop menu) or dos -s is issued. Starting MS-DOS is much
| # quicker with the .quickpc file. |
| # TEXT        |                        |
| # List of user-specified text-only applications, in addition to the standard ones
| # shipped with the Sun386i system. Running these applications from the SunOS
| # system will send output to the current window instead of opening a new DOS
| # window. |
| # BOARDS      |                        |
| # List of boards from /etc/dos/defaults/boards.pc that you want
| # to attempt to access upon opening DOS Windows. If a board is already in use,
| # it will appear as detached in the Devices submenu on the Sun386i system. In
| # this case, you can release the device from the window that owns it, and then
| # attach the device from the current window. |
As with `setup.pc`, you also can edit the `boards.pc` file; however, unlike `setup.pc`, each system should have only one copy of `boards.pc`, which affects all users on the system. If you add a device to run under MS-DOS, you must include its board name and block I/O information in the file. You must also include interrupt-level information for boards that use interrupt levels, as well as indicate whether or not the device can be shared. The `boards.pc` file included with the Sun386i system contains a list of commonly used boards included as comment lines; you can remove the comment symbols for those boards that you have and want PC applications to use.

The following table shows the AT bus I/O address spaces that DOS emulates. If you add a card in one of these address spaces, MS-DOS ignores it. If you specify an emulated address in the `boards.pc` file, the next time you open a window the system displays a message stating that the address range is already in use. You can turn off emulation for all but the hard disk by placing a comment character (#) at the start of the pertinent line in `setup.pc`.

<table>
<thead>
<tr>
<th>Address</th>
<th>MS-DOS Use</th>
</tr>
</thead>
<tbody>
<tr>
<td>1F8 – 1FF</td>
<td>Hard disk emulation</td>
</tr>
<tr>
<td>230 – 237</td>
<td>Bus mouse emulation</td>
</tr>
<tr>
<td>278 – 27F</td>
<td>Parallel port 2</td>
</tr>
<tr>
<td>378 – 37F</td>
<td>Parallel port 1</td>
</tr>
<tr>
<td>3B0 – 3BF</td>
<td>Monochrome display adapter</td>
</tr>
<tr>
<td>3D0 – 3DF</td>
<td>Color display adapter</td>
</tr>
<tr>
<td>3F0 – 3F7</td>
<td>Diskette controller</td>
</tr>
</tbody>
</table>

No two boards in the same system can have the same interrupt level. Because many boards have a factory-set interrupt level of 3, occasionally you might have to rejumper the board to set a new interrupt level, as on regular PCs. You must then also change the interrupt-level information in the `boards.pc` file before accessing the attached device. Table 7-4 shows the availability of interrupt levels for the Sun386i system. For more details about adding a board to the `boards.pc` file, refer to *Sun386i Advanced Skills*. 
7.4. Capabilities and Limitations

This section describes some MS-DOS features and limitations that you should know about. It contains sections on:

- Conversion programs for converting text files from MS-DOS to the SunOS system and vice versa
- Differences between the MS-DOS and SunOS command interpreter
- Determining the DOS Windows number to create unique file names and help avoid network collisions
- 80386 instructions supported
- Limitations such as those relating to screen height, remote port use, certain types of applications, running simultaneous versions of MS-DOS applications, interrupt rates, and space issues

Converting Between MS-DOS and SunOS Text Files

MS-DOS and the SunOS system have slightly different conventions regarding text file delimiters. Consequently, the Sun386i system includes special utilities to convert files from one set of conventions to the other. The program to convert MS-DOS files to SunOS files is called dos2unix(1); the program to convert SunOS conventions to MS-DOS conventions is called unix2dos(1). The Sun386i system contains two versions of each program, a SunOS version and an MS-DOS version, to make it easier to run both utilities from either system.

Conversion does not happen automatically. You must invoke these programs as necessary, and can do so from either the MS-DOS prompt or the SunOS prompt. Include a source file name and a destination file name on the command line.

---

Table 7-4  Interrupt Level Availability

<table>
<thead>
<tr>
<th>Interrupt Level</th>
<th>Availability</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Unavailable; used for timer emulation</td>
</tr>
<tr>
<td>1</td>
<td>Unavailable; used for keyboard emulation</td>
</tr>
<tr>
<td>2</td>
<td>Unavailable; used for interrupt controller 2 cascade</td>
</tr>
<tr>
<td>3</td>
<td>Available for board (specified in setup.pc)</td>
</tr>
<tr>
<td>4</td>
<td>Available for board, unless COM1 emulation in use</td>
</tr>
<tr>
<td></td>
<td>(specified in setup.pc)</td>
</tr>
<tr>
<td>5</td>
<td>Available for board, unless LPT2 emulation in use</td>
</tr>
<tr>
<td></td>
<td>(specified in setup.pc)</td>
</tr>
<tr>
<td>6</td>
<td>Unavailable; used for diskette emulation</td>
</tr>
<tr>
<td>7</td>
<td>Unavailable; used by built-in parallel port</td>
</tr>
<tr>
<td>8</td>
<td>Unavailable; used for real-time clock emulation</td>
</tr>
<tr>
<td>9</td>
<td>Unavailable; used by built-in serial port</td>
</tr>
<tr>
<td>10</td>
<td>Available for board</td>
</tr>
<tr>
<td>11</td>
<td>Available for board</td>
</tr>
<tr>
<td>12</td>
<td>Available for board</td>
</tr>
<tr>
<td>13</td>
<td>Unavailable; used for 8087 numeric coprocessor emulation</td>
</tr>
<tr>
<td>14</td>
<td>Unavailable; used for hard disk emulation</td>
</tr>
<tr>
<td>15</td>
<td>Available for board</td>
</tr>
</tbody>
</table>
Applications Delivery

Applications Delivery ................................................................. 137

9.1. System Software Overview ..................................................... 139
9.2. Application SunOS ................................................................. 140
  Hardware Diagnostics ............................................................... 140
  Core System ........................................................................... 140
  Optional Clusters .................................................................... 141
  Recovery Software ................................................................... 142
9.3. SunOS Developer's Toolkit ...................................................... 143
9.4. Loading and Unloading Clusters .............................................. 143
9.5. Releasing Your Software ....................................................... 144
  Installation Script ..................................................................... 144
  Start-Up Script ........................................................................ 145
  Making the Distribution .......................................................... 146A
This chapter describes software delivery — both how Sun delivers its software for the Sun386i system and the preferred method for you to deliver your applications for this system. The first part of the chapter discusses the division and distribution of system software in two major parts, and the groups of files, called clusters, constituting those parts. The last section describes the steps you should follow to enable users to easily install your applications.

9.1. System Software Overview

Sun386i system software is divided into two major sections: Application SunOS and SunOS Developer’s Toolkit. Figure 9-1 below shows the two major divisions of system software and their subsets.

<table>
<thead>
<tr>
<th>Application SunOS (unbundled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hardware Diagnostics (separate diskette)</td>
</tr>
<tr>
<td>Core System (shipped on disk)</td>
</tr>
<tr>
<td>Optional Clusters (on diskettes or tape)</td>
</tr>
<tr>
<td>Recovery Software (on diskettes or tape)</td>
</tr>
</tbody>
</table>

| SunOS Developer’s Toolkit (unbundled; loadable clusters on diskettes or tape) |

Figure 9-1  System Software Divisions

Each site receives the core system on the Sun386i system disk, as well as the Sun386i Owner’s Set documentation. All other system software and documentation is unbundled; that is, users must purchase both Application SunOS and Developer’s Toolkit to have full SunOS 4.0 functionality and accompanying documentation. Sun strongly recommends that each user site purchase at least one copy of Application SunOS. By doing so, a site receives all four pieces of Application SunOS shown in Figure 9-1.
plus the Owner's Supplement Documentation Set. The next section describes Application SunOS more fully. In addition, users can order the Developer's Toolkit, discussed in Section 9.3 on page 143.

9.2. Application SunOS

Application SunOS includes:

- Hardware Diagnostics — a set of standalone diagnostics available on diskette
- Core system — the base system, providing the ability to run most commercially available Sun and third-party applications; shipped on the Sun386i disk
- Optional clusters — additional software for capabilities such as extended mail and extended networking; available on diskettes or tape
- Recovery software — a backup version of the core system, available on diskettes or tape

Users can purchase Application SunOS on diskettes (approximately 26) or quarter-inch tape (two). In addition to the software listed above, users who purchase Application SunOS also receive the Sun386i Owner's Supplement Documentation Set. The following subsections provide details of each Application SunOS subset.

Hardware Diagnostics

The first part of Application SunOS is a set of standalone Hardware Diagnostics. These programs do not require the SunOS operating system. You should run hardware diagnostics when:

- You cannot start your system
- The system displays numerous messages indicating a hardware problem
- You have upgraded your system with a new frame buffer, memory board, or hard disk (to make sure that the new part works properly)
- Your system crashes repeatedly

For information about how to run Hardware Diagnostics and the individual tests they perform, refer to Sun386i System Setup and Maintenance.

Core System

The core system provides the minimum subset of the SunOS operating system required by every user. It is sufficient to allow users to operate most commercially available Sun and third-party applications.

The core system includes software that users always need. It uses about 19 Mbytes of disk space and is preloaded by Sun manufacturing on the formatted hard disk that comes with each system (either 91 or 327 Mbytes). There is no automated method to remove any part of the core system, since users should leave all of it on the disk.

The core system includes the groups of files listed below. The file /usr/lib/load/filesizes contains the names and sizes of files in each group.

- **base_root** — the root directory (/), which includes the kernel; system databases and start-up files; single-user mode requirements; commands such as automount(8), chown(8), fastboot(8), fasthalt(8), and reboot(8); the adb(1) and kadb(8S) debuggers; and the file /usr/lib/load/filesizes
Optional Clusters

The optional clusters included with Application SunOS are comprised of sets of related programs and files that users might need on the hard disk, in addition to the core system. Users can add individual clusters after installation by using the

**sunview** — SunView tools, icons, commands, and demos, as well as all Sun screen fonts

**boot_server** — the boot server for booting diskless Sun386i systems from Sun386i system servers, as well as the boot server enabling the Sun386i system to be a server for Sun-2, Sun-3, and Sun-4 systems

**encryption** — file encryption commands such as des(1)

**calendar** — calendar(1) program and required files

**basic_commands** — most commonly used user commands such as date(1V), grep(1V), arch(1), csh(1), passwd(1), crontab(1), and kill(1), as well as ed(1), ex(1), and vi(1) editors

**mail** — basic mail directories, files, and commands such as mail(1), biff(1), sendmail(8), and newaliases(8)

**at_commands** — commands such as at(1), atq(1), and atrm(1), for executing commands or scripts at a later time

**print_spooler** — printing commands such as lpc(8), lpd(8), lpq(1), lpr(1), and lprm(1)

**non_readonly** — configuration files, spool directories, and other nonread-only files required by optional software such as the Network File System (NFS), the print spooler, extended mail, the audit trail maintenance package, and uucp(1C) and tip(1C)

**sun_specific_commands** — commands such as click(1) and screenblank(1)

**online_help** — Spot Help and Help Viewer files, plus new kernel error messages

**key** — encryption keys for secure networking; includes chkey(1), keylogin(1), keyserv(8C), and keyenvoy(8C)

**basic_networking, rpc_base, and nfs** — contain networking software, with the exception of the boot server; includes network configuration files, daemons, and administrative and user commands such as ping(8C), rmt(8C), rcp(1C), and rlogin(1C)

**ease_of_use** — snap(1) and organizer(1) programs

**yellow_pages** — the Yellow Pages database

**dos** — MS-DOS 3.3, required to run DOS Windows

**load** — the commands required to load and unload clusters, including load(1), unload(1), loadc(1), unloadc(1), and cluster(1)
load() or loadc() commands (or the snap() administration tool), and can sub-
sequently remove them by using the unload() and unloadc() commands. (Pages
12-13 describe all four commands.) When all optional clusters are loaded, they take
about 14 Mbytes of disk space. A file that's part of the core system,
/usr/lib/load/filesizes, lists the sizes of these and all other system soft-
ware files. Sun386i System Setup and Maintenance provides more details about the
contents of these clusters, listed below.

mail_plus — extended mail commands such as from(1), vacation(1),
 uuencode(1C), uudecode(1C), and mailstats(8)

spellcheck — spell(1) program and related commands

accounting — basic accounting programs and commands such as ac(8),
 accton(8), pac(8), last(1), and lastcomm(1)

sysV_commands — basic System V commands such as uname(1V), echo(1V),
 expr(1V), cat(1V), grep(1V), sdiff(1), and chmod(1V)

advanced_admin — advanced system administration commands such as
 chroot(8), dump(8), and restore(8)

extended_commands — additional commands such as pagesize(1), trace(1),
 logger(1), and script(1)

networking_plus — extended networking utilities and commands such as
 in. fingerd, in. ftpd, and in. rwho daemons and finger(1), ftp(1C),
 rwho(1C), gettable(8C), and rpcinfo(8C) commands

audit — audit trail maintenance package, including the auditd and
 rpc.pwdauthd daemons and audit(8), audit_warn(8), praudit(8), and
 C2conv(8)

comm — uucp(1C), tip(1C), and related commands

doc_prep — text processing tools such as nroff(1), troff(1), neqn(1), and
 tbl(1), and directories required to run them

disk_quotas — quota commands such as quot(8), edquota(8), and
 quotacheck(8)

name_server — in.named, in.tnamed, and sendmail.mx daemons

man_pages — on-line man pages and man commands

plot — plotting commands such as plot(1G) and spline(1G)

old_commands — backward-compatible commands such as make(1),
 perfmon(1), clocktool(1), setkeys(1), and syslog(1)

games — on-line games, including backgammon, Boggle, and cribbage

Recovery Software

Application SunOS includes recovery software for reloading the core system, if nec-
necessary. Recovery software is available on tape and diskettes. Sun386i System Setup
and Maintenance describes how to load this software, should you need it.
9.3. SunOS Developer’s Toolkit

SunOS Developer’s Toolkit is a complement to Application SunOS, not a superset. It provides everything missing in Application SunOS needed to achieve full SunOS 4.0 functionality. In addition, the Sun386i Developer’s Toolkit Documentation Set accompanies each copy of the Developer’s Toolkit purchased. The Developer’s Toolkit is available on diskettes or quarter-inch tape. As with Application SunOS, you can add and remove Developer’s Toolkit clusters individually with the load(1) and unload(1) commands described briefly in the next section. The file /usr/lib/load/filesizes, part of the core system, lists the sizes of these and all other system software files.

SunOS Developer’s Toolkit includes the software listed below.

- **base_devel** — software development commands and utilities such as the C compiler, assembler, link editor, dbx(1); you must load this cluster to be able to use any of the Developer’s Toolkit with the exception of the help_guide cluster, which does not require base_devel

- **config** — System V files necessary to reconfigure the kernel such as config(8) and the /usr/sys directory

- **sunview_devel** — SunView development libraries required for writing window-based applications

- **help_guide** — Help Writer’s Handbook for writing on-screen help for applications (the sections Spot Help Interface and Help Viewer Interface on pages 80-94 contain much of the same information)

- **plot_devel** — libraries such as libplot.a and libplotbg.a for development plotting functions

- **proflibs** — profiled libraries (denoted by the suffix _p.a) such as libc_p.a, libm_p.a, and libcurses_p.a

- **sccs** — commands required by SCCS, the Source Code Control System

- **sysV_devel** — libraries required to port System V applications, including utilities in /usr/5bin and /usr/5lib directories

9.4. Loading and Unloading Clusters

You can issue the load(1), loadc(1), unload(1), unloadc(1), and cluster(1) commands to:

- Add one or more Application SunOS or Developer’s Toolkit clusters to the disk after installation
- Remove one or more clusters to make space for additional ones
- Display the name of a cluster containing a specified file
- Display a summary of all Application SunOS and Developer’s Toolkit clusters, including a cluster’s size and whether or not it is loaded

Pages 12-13 provide more information about these commands.
9.5. Releasing Your Software

The preferred method of releasing software on tape or diskette consists of three parts:

1. Creating an installation script.
2. Supplying a start-up script for your application, or using the `/usr/etc/start_applic` script supplied with the Sun386i system.
3. Using the `bar(1)` command (available on the Sun386i system only) to place the two files mentioned above and your application onto a formatted (format(8)) diskette, or using the `tar(1)` command to place them onto tape.

You also should provide a copyright file and installation instructions for system administrators. If you want your application to be available over the network, include the steps starting on page 97 that tell administrators to create a volume for your application and export and register it. Then users can invoke your application from any machine on the network, regardless of architecture, without having to alter their `.login` or `.cshrc` files. If you are releasing on-screen help for your application or file-type icons that users can view via the Organizer, administrators **must** follow the steps starting on page 97 to see your help files or your application’s icons.

Be sure that your installation instructions state how much space your application requires. Your instructions should also minimally tell administrators to do the following steps before installing your application.

1. Log in as superuser on the system where the application files will reside.
2. Go (`cd(1)` to the `/files<n>/vol` directory if the application will be available over the network, or to the `/usr/local` directory if it will be available only on this system.
3. Enter `mkdir` `application_name` to create an `application_name` subdirectory.
4. Enter `cd` `application_name`.
5. If the application is on diskette, enter:
   ```
   bar xvf /dev/rfd0a installation_script
   ```
   If the application is on high-density tape, enter:
   ```
   tar xvf /dev/rst8 installation_script
   ```
   If the application is on low-density tape, enter:
   ```
   tar xvf /dev/rst0 installation_script
   ```
   where `installation_script` is the name of the script you supply. (You could use additional `bar` and `tar` options, depending on the options you used to put your script on tape or diskette.)
6. Enter `installation_script`.

The final step will invoke the installation script that you supply, which must perform the steps listed in the next section.

**Installation Script**

Installation scripts will vary between applications in content and complexity. The basic things an installation script should do are shown below. For additional information on what installation scripts and system administrators **must** do if your application includes on-screen help files or Organizer icons for your files, see page 97.

Revision B, June 1988
Pages 207-208 include a description of the directory hierarchy suggested for applications.

Instruct administrators not to perform these steps when anyone is adding a diskless system to the network or is creating a user account on it through New User Accounts or SNAP. This is to avoid possible corruption of the /etc/exports file, used by both procedures and modified in the steps below.

1. Load the application files into the current directory. For diskettes, use the command:
   ```
   bar xvf /dev/rfd0a directories
   ```
   For high-density tapes, use the command:
   ```
   tar xvf /dev/rst8 directories
   ```
   For low-density tapes, use the command:
   ```
   tar xvf /dev/rst0 directories
   ```
   The tar or bar options could be different, depending on the ones you used to put your files on tape or diskette.

2. If licensing issues permit, make your application available over the network by exporting it, as shown by the following steps. (Alternatively, you could have a system administrator perform these steps; see pages 97-98.)
   
   a. Make sure that the /export/vol directory exists:
      ```
      mkdir /export/vol > /dev/null 2>&1
      ```

   b. Create a symbolic link using the command format:
      ```
      ln -s '/bin/pwd' /export/vol/application_name
      ```

   c. Enter the following line to add an entry for the application in the /etc/exports file:
      ```
      echo "/export/vol/application_name -ro" >> /etc/exports
      ```

   d. Enter the following command to export the application:
      ```
      exportfs /export/vol/application_name
      ```

   If your application can only run on systems where it is installed, then your installation instructions should tell administrators to load your application into /usr/local on a system that is not a server. Your instructions should also tell administrators not to export your application.

   You might also want to include an interactive portion in the script that queries an administrator about the configuration and about the architecture (if the release includes binaries for several different architectures).

   **NOTE** Be sure to use only relative path names in your script and application. Regardless of where your application is initially installed, administrators are likely to move it elsewhere to suit their needs. Where a complete path name is necessary, use the $application_name_ROOT environment variable described on page 207.

**Start-Up Script**

The primary benefit of using a start-up script to invoke your application is that system administrators can then register your application, enabling anyone on a network or within a specified netgroup (see the netgroup(5) man page) to invoke your
application without modifying /etc/fstab for each system on the network, or
.login or .cshrc files. Users also need not know the path to your application,
or the architecture of the machine where your files reside.

Smaller applications consisting of one or two files do not need a start-up script, pro­
vided that the files are accessed from either /vol/local/bin. arch or
/usr/local/bin. (arch can be either sun386, sun2, sun3, or sun4.) The
/vol/local and /usr/local directories are all included in user’s default
$PATH values.

If you do include a start-up script, you must:

• Use the $application_name_ROOT environment variable in your applica­
tion binaries if you refer to your files. If you program in C, use the
getenv(3) command in the format
getenv("application_name_ROOT"). Page 207 provides more informa­
tion about the $application_name_ROOT variable.

• Adhere to the application directory hierarchy described on pages 207-208.

In addition, to make your application available on a network, your installation
instructions must tell system administrators to:

1. Export your application, if your script does not do this for them.
2. Create a volume for your application.
3. Copy your start-up script or /usr/etc/start_applic to
   /vol/local.master/bin, giving the script the name of your applica­
tion. (/vol/local.master/bin is included in every user's $PATH by
default.)

Chapter 6 delineates the steps a system administrator must perform, whether your
application will be available on a network or only on one system, starting on page
96. The next section describes the /usr/etc/start_applic script that is sup­
plied with the Sun386i system.

The /usr/etc/start_applic script is a short, generic shell script that an
administrator can copy or link into either
/vol/local.master/bin/application_name (for applications available on a net­
work) or /usr/local/bin/application_name (for applications available only on
the systems where they are installed). The script ensures that the $PATH,
$application_name_ROOT, and $LANG variables are correctly set when the applica­
tion runs.

In most cases, the /usr/etc/start_applic script that is provided with the
Sun386i system should meet your start-up script needs. Your application must
include an executable binary containing the application name in the location
$application_name_ROOT/bin. arch/application_name if your application uses the
/usr/etc/start_applic start-up script.

If your application does not have an application_name binary in the bin. arch subdi­
rectory, you can create a customized start-up script. If you create a script, it should
be a superset of the contents of /usr/etc/start_applic.

/usr/etc/start_applic locates and invokes the application, if possible. If it
cannot start the application, it displays one or more error messages to try to explain
the difficulty. The start_applic(8) man page contains additional information.
Making the Distribution

For distributions on diskette, be sure to format each diskette with `fddformat` for high-density diskettes or `fddformat -L` for low-density diskettes before copying your files to them. (The `fddformat(8)` man page contains more information.) Then use the `bar(1)` command to put your files on the diskette. You should also use the `bar` command if any of your application files exceeds the size of one diskette or tape, since `bar` can copy information across multiple volumes. See the `bar(1)` man page for details.

The `bar` command is available for loading tapes and diskettes only on Sun386i systems. Therefore, unless one of your files is too large for one cartridge tape, use the `tar(1)` command for tape distributions. The `tar(1)` man page provides specifics.
Figure 10-1  U.S. Keystation Map

Figure 10-2  International Keystation Map
### Floating Accent Key

You can create accent characters with the floating accent key available on international keyboards. The floating accent key works similarly to the \texttt{Compose} key, in that you press the floating accent key and then the character that you want to accent. You can accent both lower- and uppercase characters; to accent the latter, press the floating accent key, the \texttt{Shift} key, and then the character that you are accenting.

### Native-Language Messages

To aid in the translation process, the Sun386i system manages external message libraries and routines for selecting and accessing system messages and on-screen help text, as described below.

### System Messages

System messages are those originating in the SunOS kernel and system daemons. The \texttt{syslogd(8)} daemon intercepts error messages and uses them as keys into a message...
8-Bit (byte) General Registers

%al  Low byte of %ax register
%ah  High byte of %ax register
%cl  Low byte of %cx register
%ch  High byte of %cx register
%dl  Low byte of %dx register
%dh  High byte of %dx register
%bl  Low byte of %bx register
%bh  High byte of %bx register

16-Bit (word) General Registers

%ax  Low 16-bits of %eax register
%cx  Low 16-bits of %ecx register
%dx  Low 16-bits of %edx register
%bx  Low 16-bits of %ebx register
%sp  Low 16-bits of the stack pointer
%bp  Low 16-bits of the frame pointer
%si  Low 16-bits of the source index register
%di  Low 16-bits of the destination index register

32-Bit (long) General Registers

%eax 32-bit general register
%ecx 32-bit general register
%edx 32-bit general register
%ebx 32-bit general register
%esp 32-bit stack pointer
%ebp 32-bit frame pointer
%esi 32-bit source index register
%edi 32-bit destination index register

Segment Registers

%cs  Code segment register; all references to the instruction space use this register
%ds  Data segment register, the default segment register for most references to memory operands
%ss  Stack segment register, the default segment register for memory operands in the stack (i.e., default segment register for %bp, %sp, %esp, and %ebp)
%es  General-purpose segment register; some string instructions use this extra segment as their default segment
%fs  General-purpose segment register
%gs  General-purpose segment register
This section describes the SunOS 386 instruction syntax. Refer to page 178 for the differences between the SunOS 386 and the Intel 386 assemblers.

Because the assembler assumes it is generating code for a 32-bit segment, it also assumes a 32-bit address and automatically precedes word operations with a 16-bit data prefix byte.

This section uses the following notational conventions:

- The mnemonics are expressed in a regular expression-type syntax. Alternatives separated by a vertical bar (|) and enclosed within square brackets ([]) denote that you must choose one of them. Alternatives enclosed within curly braces ({}) denote that you can use one or none of them. The vertical bar separates different suffixes for operators or operands. For example, `imm[8|16|32]` indicates that an 8-, 16-, or 32-bit immediate value is permitted in an instruction.

- `imm[8|16|32|48]` — an immediate value. You define immediate values using the regular expression syntax previously described (see also Expressions and Immediate Values on page 182). If there is a choice between operand sizes, the assembler will choose the smallest representation.

- `reg[8|16|32]` — a general-purpose register, where each number indicates one of the following:
  - 32: (%eax), (%ecx), (%edx), (%ebx), (%esi), (%edi), (%ebp), (%esp)
  - 16: (%ax), (%cx), (%dx), (%bx), (%si), (%di), (%bp), (%sp)
  - 8: (%al), (%ah), (%cl), (%ch), (%dl), (%dh), (%bl), (%bh)

- `mem[8|16|32|48]` — a memory operand; the 8, 16, 32, and 48 suffixes represent byte, word, doubleword, and inter-segment memory address quantities, respectively.

- `r/m[8|16|32]` — a general purpose register or memory operand; the operand type is determined from the suffix. They are: 8 = byte, 16 = word, and 32 = doubleword. The registers for each operand size are the same as `reg[8|16|32]` above.

- `creg` — a control register; the control registers are: %cr0, %cr2, or %cr3.

- `dreg` — a debug register; the debug registers are: %db0, %db1, %db2, %db3, %db6, %db7.

- `sreg` — a segment register; the segment registers are: %cs, %ds, %ss, %es, %fs, and %gs.

- `treg` — a test register; the test registers are: %tr6 and %tr7.

- `cc` — condition codes; the 30 condition codes are:
  - a above
  - ae above or equal
  - b below
  - be below or equal
  - c carry
  - e equal
File System Layout

C.1. Terms ............................................................................................................ 195
C.2. Layout Overview .......................................................................................... 196
    System Disk ................................................................................................ 196
    Additional Disks ......................................................................................... 197
C.3. / File System ............................................................................................... 197
C.4. /usr File System ........................................................................................ 200
C.5. /files<n> File System ............................................................................. 203
C.6. /export Directory ..................................................................................... 204
C.7. /vol Directory ............................................................................................ 206
C.8. Application Directory Structure .................................................................. 207
This appendix describes the file system layout for the Sun386i system. Much of this structure is similar to that of the SunOS 4.0 system on other architectures. The 4.0 layout is intended to make it easier for a single server to support clients of different architectures. The Sun386i layout also provides a standard place to mount additional disks and makes it possible for users to access network files and applications without knowing the location of files or the architecture of the workstation being used, and without changing .login or .cshrc files. In addition, the Sun386i layout consolidates all of the free disk space onto one partition.

Some file system differences also exist between Sun386i systems and other Sun systems to accommodate the division of Sun386i system software. The Sun386i system groups much of system software into related files and programs called clusters, which users optionally can add to their systems at any time (Chapter 9 contains details). For compatibility between systems, and because previously existing programs expect to find files in their traditional directories, some directories now contain symbolic links to directories that contain the actual files.

If you'll be distributing software for the Sun386i system, be sure to review the last section, C.8, which describes the preferred directory structure for releasing software for this workstation.

C.1. Terms

The Sun Network File System (NFS) allows any computer with a local disk to act as a file server by exporting its file systems to clients on a network. The client computers may themselves be file servers of other file systems.

The Yellow Pages (YP) is a distributed network database. Key information about the systems and users on the network is stored in the YP database on the master server and slave servers. The master server keeps the master copy of the database, using it to update identical copies on slave servers. The YP is stored on the master server and all the slave servers to ensure the availability of the database in case a server goes down. However, the master server must be running for the updates to YP to occur. Sun386i Advanced Administration discusses NFS and YP in more detail.

The automounter (automount(8)) is a daemon that automatically and transparently mounts an NFS file system at a temporary mount point whenever a file or directory within that system is opened. The mounted file system is made available using a symbolic link to the mount point. Sun386i Advanced Administration and the automount(8) man page contain more information.
Several Sun386i 4.0 directories on diskful systems are *loopback mounted*. Files that appear to be in a loopback-mounted directory on diskful systems actually reside on another locally mounted directory. For example, on a diskful system files in `/tmp` really reside in `/files/tmp/localhost` because `/tmp` is a loopback mount to `/files/tmp/localhost` on diskful systems. On diskless systems, `/tmp` is in the client's root directory, which is mounted from its server. See the lofs(4S) man page for additional information about loopback mounts.

### C.2. Layout Overview

The SunOS 4.0 file system layout:
- Provides easier maintenance of servers and clients
- Enables easier mixing of remote and local file systems
- Provides cleaner support of multiple architectures
- Provides a hierarchy that accounts for growth, enabling users to mount additional disks without affecting file names
- Minimizes disruption to existing programs when files are moved
- Minimizes symbolic link confusion
- Minimizes user confusion when they log in to different systems

Sun386i software is divided among three primary file systems: `/root`, `/usr`, and `/files`. In addition, the `/vol` and `/home` directories help to ensure that the file system looks the same to all network users. These file systems and directories are described briefly in the next section.

#### System Disk

The disk that the system boots from is called the *system disk*. The standard system disk layout consists of the following special device files and partitions:

- `/dev/roota` — contains the root (`/`) file system
- `/dev/rootb` — contains the system's swap area
- `/dev/rootg` — contains the `/usr` file system
- `/dev/rooth` — contains the `/files` file system

The default `/etc/fstab` file contains entries to mount the corresponding file systems using the partitions listed above. This enables users to boot systems from any disk without modifying `/etc/fstab`.

The standard Sun386i file systems and directories are briefly described below. Subsequent sections detail the contents of each one.

#### / (root) File System

/ (root) is the major SunOS file system, located at the top of the hierarchical file system tree. It contains machine-specific files and directories crucial for system operation, such as the kernel, a device directory listing the equipment for the configuration, and programs used for booting the multiuser version of the operating system. The contents of / is described on the following page.

#### /usr File System

/`usr` contains executable commands, system programs, and library routines, as well as some executables that were formerly under `/` (such as system administration programs). This is a read-only file system to ensure that its contents are identical.
throughout the network and to enable network-wide mounting and sharing. Because
/usr is read-only, users see the same files regardless of where they log in. /usr is
intentionally very full. The contents of /usr is shown on page 200.

/files File System
/files contains free space remaining after allocation of the root, swap, and /usr
partitions. /files contains home directories, unbundled and third-party applications,
optionally loaded Application SunOS and Developer's Toolkit clusters, and
root, swap, and dump directories for diskless clients. The contents of /files is
delineated on page 203. Disks added to the system are mounted on /files<n>, as
described in the next section, Additional Disks.

/home Directory
/home is used in conjunction with the automounter (automount(8)) to provide
transparent access to a user's home directory, regardless of where the directory
resides on the network. /home is described further as part of the / file system on
the next page.

/export Directory
/export contains symbolic links to the local files and directories that diskful sys-
tems export to other machines on the network. Only the links, not the directories
themselves, are stored in /export. Other systems on the network cannot see the
actual location of exported directories. Pages 204–206 provide more information on
/export.

/vol Directory
/vol is an automount(8) directory for volumes. A volume is a collection of related
files dedicated to the same function, such as data files or all files required to run a
Sun unbundled or third-party application. /vol is described more fully on page
206.

Additional Disks
Additional disks added to the Sun386i system have one partition, sdnc, where n is
the SCSI unit number. If the disk in the expansion unit is the system disk, n is 0. If
the system unit houses the system disk, n is 2. sdnc provides access to the entire
disk. Each additional disk is mounted on /dev/sdnc as /files<x>, where x indicates
the order of the disk added (/files1, /files2, and so on). x and n are not related.

C.3. / File System

/ contains the following files and directories:

<table>
<thead>
<tr>
<th>bin</th>
<th>files&lt;n&gt;</th>
<th>mnt</th>
<th>tmp</th>
<th>vmunix</th>
</tr>
</thead>
<tbody>
<tr>
<td>boot</td>
<td>home</td>
<td>net</td>
<td>tmp</td>
<td>vol</td>
</tr>
<tr>
<td>dev</td>
<td>kadb</td>
<td>sbin</td>
<td>usr</td>
<td></td>
</tr>
<tr>
<td>etc</td>
<td>lib</td>
<td>sys</td>
<td>var</td>
<td></td>
</tr>
<tr>
<td>export</td>
<td>lost+found</td>
<td>tftpboot</td>
<td>VERSION</td>
<td></td>
</tr>
</tbody>
</table>

Starting with Release 4.0, /bin is a symbolic link to /usr/bin/.

/boot

boot is the program used to load the SunOS operating system. Never alter or
remove this program from a disk.
/dev is the device directory, which contains all device files (also called device nodes) such as /dev/rst0 (quarter-inch tape drive), /dev/ttya (serial port), or /dev/pp0 (parallel port), for a particular configuration.

/etc contains system-specific data files and subdirectories primarily used by system administrators; includes the files created during installation and some machine-specific MS-DOS files in /etc/dos/.

/export is described in Section C.6, starting on page 204.

/files is the mount point for the /files file system, described in Section C.5 starting on page 203.

/home is an automounter directory that provides automatic access to home directories for all users on the network. By default, users' home directories are stored in /files/home/groupname/username on various systems, but passwd(5) Yellow Pages (YP) entries for each user specify the home directory path as /home/username. The automounter (automount(8)) takes references to /home/username and uses the auto.home YP map to return symbolic links to the home or -username directory. /home is shipped empty; /home/username does not exist as part of the file system on disk, but rather is created only after an automount reference is made to it.

If the auto.home entry indicates that the home directory is on a remote system, the automounter creates a temporary mount point under /tmp_mnt and uses this point to mount the remote directory onto the local system via NFS. The automounter returns a symbolic link to the mount point.

If the home directory is on the local system, the automounter returns a symbolic link to the directory. For more information on the automounter, see Sun386i Advanced Administration, and the auto.home(5) and automount(8) man pages.

/kadb

kadb(8) is the kernel debugger program.

/lib

/lib is a symbolic link to usr/lib.

/lost+found

/lost+found is usually empty. However, if / becomes damaged, the file system check program (fsck(8)) places links to any files that it cannot link elsewhere in this file system in /lost+found.

/mnt

/mnt is a mount point for temporarily mounting systems with the mount(8) command.

/net is an automount(8) point available on networked systems that is used by the Organizer and the SNAP backup facility. Experienced users can also use /net to access directories on remote NFS diskful systems, although /home and /vol are preferred. Before using /net, make sure the system is an NFS file server (diskful system) and the path name is exported (/net/hostname/export/pathname).
/sbin contains executable files necessary to check and mount the /usr file system and to bring up a multiuser system at boot time:

- /sbin/fsck — checks and repairs the file system
- /sbin/init — performs process control initialization
- /sbin/mount — mounts file systems
- /sbin/netconfig — configures the network
- /sbin/reboot — reboots the system
- /sbin/sh — standard command interpreter

/sys is a symbolic link to ./usr/share/sys/.

tftpboot contains the files necessary to boot diskless clients on the network.

/tmp holds files temporarily; utilities such as cc(1) and ar(1) create temporary data files in /tmp. All files in /tmp, with the exception of subdirectories, are deleted each time the system is rebooted. On Sun386i diskful systems, this is loopback mounted onto /files/tmp/localhost. For diskless systems, /tmp is in the client's root (mounted from the server).

tmp_mnt is the directory that the automounter (automount(8)) uses to make mount points for temporary file systems. Do not add files to or remove files from this directory.

/usr is the mount point for the /usr file system, described on the following page.

/var contains the following directories and symbolic links. On Sun386i diskful systems, /var is loopback mounted onto /files/var/localhost. On diskless systems, /var is in the client's root (mounted from the server).

- /var/adm — contains system accounting and log files
- /var/crash — is reserved for kernel core dumps of servers and standalone systems if they crash; shipped empty
- /var/dos — reserved for MS-DOS files for root (the same files in ~username/pc for other users); shipped empty
- /var/log — directory for log files; shipped empty
- /var/preserve — holds files saved by the vi and ed editors if the system crashes
- /var/recover — directory for crash recovery scripts (on the Sun386i system only); shipped empty
- /var/spool — contains files used for printing and other spooling functions
- /var/sysex — directory where the System Exerciser writes its temporary and log files. The System Exerciser runs under the SunOS system and verifies operation of the total system, including operating system software (on Sun386i systems only).
- /var/tmp — contains temporary files placed here by programs; unlike /tmp, files in this directory are not deleted when you reboot the system
- /var/yp — directory containing Yellow Pages databases
VERSION is a text file specifying the version of the root file system.

/vmunix

/vmunix is the SunOS system kernel.

/vol

/vol is an automount(8) directory described in Section C.7.

C.4. /usr File System

/usr is mounted read-only and shared. It contains architecture-specific executables and libraries, including the files and directories:

<table>
<thead>
<tr>
<th>5bin</th>
<th>dict</th>
<th>local</th>
<th>share</th>
</tr>
</thead>
<tbody>
<tr>
<td>5include</td>
<td>dos</td>
<td>lost+found</td>
<td>spool</td>
</tr>
<tr>
<td>5lib</td>
<td>etc</td>
<td>man</td>
<td>stand</td>
</tr>
<tr>
<td>adm</td>
<td>games</td>
<td>mdec</td>
<td>sys</td>
</tr>
<tr>
<td>bin</td>
<td>hosts</td>
<td>old</td>
<td>sysex</td>
</tr>
<tr>
<td>boot</td>
<td>include</td>
<td>pub</td>
<td>tmp</td>
</tr>
<tr>
<td>cluster</td>
<td>lib</td>
<td>sccs</td>
<td>ucb</td>
</tr>
</tbody>
</table>

VERSION

/user/5bin

/user/5bin contains symbolic links to UNIX System V binary files, stored in cluster/devel/sysV_devel/5bin/ if the sysV_devel cluster is loaded.

/user/5include

/user/5include is a symbolic link to cluster/devel/sysV_devel/5include/, which contains UNIX System V include files if the sysV_devel cluster is loaded.

/user/5lib

/user/5lib contains symbolic links to UNIX System V libraries, stored in cluster/devel/sysV_devel/5lib/ if the sysV_devel cluster is loaded.

/user/adm

/user/adm is a symbolic link to ../var/adm.

/user/bin

/user/bin contains basic SunOS operating system commands, including those formerly located in /bin, such as ls(1V), cat(1V), and mkdir(1).

/user/boot

/user/boot is a directory that contains symbolic links to files in ../../sbin.

/user/cluster

/user/cluster is loopback mounted onto /files/cluster on Sun386i diskful systems. (On diskless Sun386i systems, /usr/cluster is a mount point.) /usr/cluster contains all of the optional clusters added to Sun386i systems, in the directories:

- /usr/cluster/appl — mount or load points for Application SunOS clusters added to the system
- /usr/cluster/devel — mount or load points for SunOS Developer’s Toolkit clusters added to the system

NOTE

Do not use /usr/cluster/<appl devel> path names in programs, because the location of clusters could change in subsequent releases. Instead, use the link names that resolve to /usr/cluster/<appl devel>, such as /usr/dict (on the next page) and others described later in this appendix.

Revision B, June 1988
/usr/dict is a link to cluster/appl/spellcheck/dict/, which is a database that contains English language spelling lists used by the spell(1) spelling checker, if the optional cluster spellcheck is loaded; shipped empty.

/usr/dos contains MS-DOS commands and files.

/usr/etc contains the commands and files used for system administration and maintenance.

/usr/games is a symbolic link to cluster/appl/games/games/, which exists only if the optional games cluster is loaded.

/usr/hosts is a symbolic link to old/hosts.

/usr/include contains links to all of the standard include (header) files used in C programs; these files, traditionally named with a .h extension, contain definitions of useful constants and macros. Include files are either in the sys subdirectory or in ../cluster/devel/base_devel/include if the base_devel cluster is loaded. /usr/include contains the subdirectories shown below:

- /usr/include/images — contains SunView icons
- /usr/include/make — contains a default make file

/usr/lib and its subdirectories contain more than 100 files and links to files used by SunOS utilities and files formerly located in /lib (which is now a symbolic link to /usr/lib).

/usr/lib includes the subdirectory /usr/lib/load, which contains the cluster size and file database used by the load(1), loadc(1), unload(1), unloadc(1), and cluster(1) commands; also contains the filesizes file which lists the names and sizes of files within each cluster (on the Sun386i system only).

/usr/local is loopback mounted to /export/local/arch on diskful systems, where arch can be either sun2, sun3, sun4, or sun386. On diskless systems, /usr/local is an NFS mount to /export/local/arch on the boot server. /usr/local contains binaries that only the local system can use and sharable data files in the subdirectories:

- /usr/local/bin — contains binaries and shell scripts for the local architecture; included in every user’s $PATH by default
- /usr/local/lib — contains libraries for the local architecture
- /usr/local/application_name — reserved for installation of third-party software for access only by the local system

/usr/lost+found is usually empty. However, if /usr becomes damaged, the file system check program (fsck(8)) places links to any files that it cannot link elsewhere in this file system in /usr/lost+found. If this happens, you’ll probably have to reload the core system, as described in Sun386i System Setup and Maintenance, as well as any optional clusters that you had added.

Revision B, June 1988
/usr/man is a symbolic link to share/man/, described later in this section.

/usr/mdec is a symbolic link to cluster/appl/advanced_admin/mdec/, which exists only if the optional advanced_admin cluster (containing boot blocks and the install boot program for the Sun386i system) is loaded.

/usr/old is a symbolic link to cluster/appl/old_commands/old, which exists only if the optional old cluster is loaded. The old cluster contains commands that have been phased out but retained in this release for compatibility.

/usr/pub is a symbolic link to cluster/appl/doc_prep/pub/, which contains data files used in formatting and printing if the optional doc_prep cluster is loaded.

/usr/sccs is a symbolic link to cluster/devel/sccs/sccs, which exists only if the optional sccs cluster is loaded.

/usr/share contains architecture-independent sharable files, shown below:

- /usr/share/lib — contains tab set, termcap, time zone, and New User Accounts login screens; also contains the terminfo file, which is a link to terminal information in
  
  ../..//cluster/appl/sysV_commands/share/lib/terminfo/

  if the optional sysV_commands cluster is loaded

- /usr/share/man — symbolic link to
  
  ../..//cluster/appl/man_pages/share/man/, which exists only if the optional man_pages cluster is loaded

- /usr/share/messages — contains messages files for utilities shipped on the Sun386i disk

- /usr/share/src/sun/sun/suntool — symbolic link to source code for some SunView programs in
  
  ../..//cluster/devel/sunview_devel/share/src/sun/sun tool,

  which exists only if the optional sunview_devel cluster is loaded

- /usr/share/sys — symbolic link to
  
  ../..//cluster/devel/config/share/sys/, which contains the files needed to build custom kernels, and exists only if the optional config cluster is loaded

/usr/spool is a symbolic link to ../var/spool/.

/usr/stand is a symbolic link to ../stand/.

/usr/sys is a symbolic link to share/sys/.

/usr/sysex contains System Exerciser executable files.

/usr/tmp is a symbolic link to ../var/tmp/.

/usr/ucb contains commands that originated with the Berkeley UNIX system (ucb is an acronym for University of California at Berkeley).
/usr/VERSION

/usr/VERSION is a text file specifying the version of this /usr file system.

C.5. /files<n> File System

/files<n> could contain the directories shown in this section (not all of these will exist on disks added to the expansion unit). /files is the name of this file system on the system disk. The name of this file system on the first additional disk added to the system is /files1, for the second disk added /files2, and so on.

<table>
<thead>
<tr>
<th>cluster</th>
<th>help</th>
<th>lost+found</th>
<th>tmp</th>
</tr>
</thead>
<tbody>
<tr>
<td>dump</td>
<td>home</td>
<td>root</td>
<td>var</td>
</tr>
<tr>
<td>exec</td>
<td>local</td>
<td>swap</td>
<td>vol</td>
</tr>
</tbody>
</table>

/files/cluster

/files/cluster/<arch>.<OS release> contains optional clusters added to the Sun386i disk, in the directories shown below. <arch> can be either sun2, sun3, sun4, or sun386 and <OS release> has the format SunOS4.0.0, SunOS4.0.1, SunOS4.1.0, and so on.

- /files/cluster/<arch>.<OS release>/appl/cluster_name — contains Application SunOS clusters added to the system
- /files/cluster/<arch>.<OS release>/devel/cluster_name — contains SunOS Developer's Toolkit clusters added to the system

/files<n>/dump

/files<n>/dump is reserved for kernel core dumps of diskless clients if they crash; shipped empty and not used by default.

/files<n>/exec

/files<n>/exec contains the native executables (typically symbolic links to /usr file systems) of each Sun workstation architecture and each SunOS release on the server's disk.

/files/help

/files/help contains links to the on-screen help files supplied with the Sun386i system in /usr/lib/help. (These files are accessed through /vol/help.) If you delete anything in this directory, you will not be able to access any of the Sun386i on-screen help files. The only file in this directory that is not a link is /files/help/language/USA-English/Top_Level, which you can edit to add help handbooks that are not supplied with the Sun386i system.

/files<n>/home

/files<n>/home is reserved for the home directories of each user on a server (/files<n>/home/groupname/username).

/files<n>/local

/files<n>/local contains the subdirectories:

- /files/local/arch — contains the local system's /usr/local (a loopback mount from /export/local/arch) and is shared with any diskless client of the same architecture
- /files/local/other_arch — contains the /usr/local used by diskless clients that have an architecture different from the server

/files<n>/lost+found

/files<n>/lost+found is usually empty. However, if /files<n> becomes damaged, the file system check program (fsck(8)) places links to any files that it cannot link elsewhere in this file system in /files<n>/lost+found.
/files<n>/root

/ files<n>/root is reserved for root directories for all diskless clients of a server (/ files<n>/root/hostname).

/ files<n>/swap

/ files<n>/swap is reserved for individual swap areas for all diskless clients of a server (/ files<n>/swap/hostname).

/ files<n>/tmp

/ files<n>/tmp contains the localhost directory. / tmp is loopback mounted to / files<n>/tmp/localhost on diskful systems; refer to the description for / tmp earlier in this appendix. To move the tmp directory to another disk (from / files to / files1, for example), edit the /etc/fstab file. The fstab(5) man page provides details.

/ files<n>/var

/ files<n>/var contains the localhost subdirectory. / var is loopback mounted to / files<n>/var/localhost on diskful systems; refer to the description for / var earlier in this appendix. To move the var directory to another disk (from / files to / files1, for example), edit the /etc/fstab file. The fstab(5) man page provides details.

/ files<n>/vol

/ files<n>/vol is reserved for third-party applications that will be available over the network.

/ files<n>/vol.local

/ files<n>/vol.local is on the server that contains / vol/local (the Yellow Pages master by default). It is reserved for shell scripts and architecture-dependent executable files.

- bin — designed to contain application start-up scripts, or links that point to /usr/etc/start_app Ric for each application that has its own volume
- bin.<arch> — contains architecture-dependent executable files, where arch can be either sun2, sun3, sun4, or sun386

C.6. /export Directory

/export contains symbolic links to local directories that diskful systems export to other machines on the network. Many of these links already exist on the system, and others are created when performing certain administrative functions via SNAP or New User Accounts. In both cases, the system edits the /etc/exports file to include information on exported directories.

Page 97 and the exports(5) and exportfs(8) man pages contain more information about exporting directories.

The leaf nodes (final components of path names) that the system typically exports are shown below.

/export/cluster

/export/cluster contains an <arch>.<OS release> directory that contains a link to ..../files/cluster/<arch>.<OS release>/, which contains clusters added to Sun386i systems. <arch> can be either sun2, sun3, sun4, or sun386 and <OS release> has the format SunOS4.0.0, SunOS4.0.1, SunOS4.1.0, and so on.

/export/dump

/export/dump on a server is reserved for manually setting up dump directories for diskless clients. By default, diskless clients dump to the swap partition.
/export/exec

/export/exec contains a symbolic link for each software architecture of the SunOS system loaded on this workstation. Each link points to the particular release’s location:

/export/exec/arch is a symbolic link to /export/exec/<arch>. <OS release>, which is a symbolic link to /usr if the server is running the same <arch>. <OS release> as the /exec being exported; if this is not the case, then /export/exec/<arch>.<OS release> is a symbolic link to /files<>/exec/<arch>.<OS release>

where <arch> can be either sun2, sun3, sun4, or sun386 and <OS release> has the format SunOS4.0.0, SunOS4.0.1, SunOS4.1.0, and so on.

Diskless clients mount /usr from their boot server, using the entry that Automatic System Installation places in the bootparams Yellow Pages map:

    usr = server:/export/exec/<arch>.<OS release>

/export/help

/export/help is a symbolic link to ../files/help/.

/export/home

If this system has a disk with users’ home directories, /export/home contains symbolic links to each user’s home directory:

/export/home/groupname/username is a symbolic link to /files<>/home/groupname/username

/export/local

/export/local/arch points to /usr/local for systems of the architecture supported. arch can be either sun2, sun3, sun4, or sun386.

/export/root

If this system is a boot server for diskless clients, /export/root contains symbolic links to each client system’s root directory:

/export/root/client_systemname is a symbolic link to /files<>/root/client_systemname

Diskless clients mount / from their boot server, using the entry that Automatic System Installation places in the bootparams Yellow Pages map:

    root = server:/export/root/client_systemname

/export/swap

If this system is a boot server for diskless clients, /export/swap contains symbolic links to each client system’s swap file:

/export/swap/client_systemname is a symbolic link to /files<>/swap/client_systemname

/export/tmp

/export/tmp contains the localhost file, which is a symbolic link to ..../files/tmp/localhost.

/export/var

/export/var contains the localhost file, which is a symbolic link to ..../files/var/localhost.

Revision B, June 1988
/export/vol

/export/vol is a standard place from which to export volumes. By default /export/vol on the master server contains two subdirectories:

- /export/vol/help — a symbolic link to /files/help
- /export/vol/local — a symbolic link to /files/vol.local

On other systems /export/vol is designed to contain exported volumes.

C.7. /vol Directory

A volume is a collection of related files dedicated to the same function, such as all files required to run a third-party or unbundled application, or all data associated with a particular project. Volumes are attached to the Sun386i file system by the automounter (automount(8)). After installation on a Sun386i network, the Sun386i system minimally has these four volumes:

- /vol/help — the default help directory (read only) for on-screen help files
- /vol/help.master — a writable copy of /vol/help
- /vol/local — a network-wide, multiple-architecture directory (read only) for accessing programs; contains a /bin subdirectory for shell scripts and a bin. arch subdirectory for architecture-specific binaries (arch can be either sun2, sun3, sun4, or sun386)
- /vol/local.master — a writable copy of /vol/local

To create a volume for an application, an administrator must export the application as /export/vol/application_name and include an entry in the /etc/auto.vol file on the YP master with the format:

application_name [-mount options] system:/export/vol/application_name

where -mount options might include either -ro or -ro, secure, since volumes that you create are mounted with read-write access by default. (/vol/help and /vol/local are read-only.) Then an administrator must rebuild the auto.vol YP map by issuing the commands:

cd /var/yp; make

The automounter (automount(8)) takes references to /vol/application_name and uses the entry corresponding to application_name in the auto.vol Yellow Pages map to mount the volume on a temporary mount point under /tmp_mnt. /vol subdirectories do not exist as part of the file system on disk, but rather are created only after automount(8) references to them are made. (Sun386i Advanced Administration and the automount(8) man page describe the automounter in more detail.)

If the auto.vol entry indicates that the volume is on a remote system, the automounter creates a temporary mount point under /tmp_mnt and uses this point to mount the remote volume onto the local system via NFS. The automounter returns a symbolic link to the mount point.

If the volume is on the local system, the automounter returns a symbolic link to the volume.

Revision B, June 1988
If the application has been exported, is accessible from a volume, and has its start-up script in /vol/local.master/bin (which registers it), any user on the network can execute the application from a workstation of any supported architecture simply by entering application_name; users’ .login and .cshrc files need no modification, and the application’s full path name is not needed. Section 9.5 and Sun386i Advanced Administration provide details.

C.8. Application Directory Structure

This section describes the preferred subdirectories that you should use to release your applications. You should use this structure because many administrators will want to make your application available on a network (provided that licensing issues permit such access), which could consist of systems with different architectures.

In addition, system administrators will install and potentially move your application to wherever is best for them. That is why your application files should use the environment variable $application_name_ROOT in path names instead of using absolute paths. The $application_name_ROOT variable provides one place for all of your application files, and ensures that your application will work, regardless of its location. If your application uses the $application_name_ROOT variable as suggested, then you must also include a start-up script with your application to correctly set the environment variable. This can be either the one supplied with the Sun386i system, /usr/etc/start_applic, or one that you provide. Chapter 9 contains more information on start-up scripts.

To make applications available on a network, administrators should install your application in /files<n>/vol/application_name, and then export and create a volume for it, as described starting on pages 96-98. Using this scheme enables your application to work even when administrators (or users with workstations) move it to a different location, and eliminates the need for users to alter their .login and .cshrc files to use your application. Administrators should install your application under the /usr/local/application_name directory for applications that can only be accessed from one system (generally for licensing reasons).

You should include the following relevant subdirectories of $application_name_ROOT for your application. Note that all subdirectories not explicitly tagged with a processor architecture are shared among all processor architectures.

bin.<arch>
This directory is for architecture-dependent binary files, where arch can be either sun2, sun3, sun4, or sun386. Your application must include an executable binary containing the application name in the format $application_name_ROOT/bin.arch/application_name if your application uses the /usr/etc/start_applic start-up script.

bin
This directory is for shell scripts or other sharable, interpreted programs for the application.
share
Place architecture-independent files in share, using the subdirectories:

- data — for miscellaneous data files
- fonts — for font files
- icons — for SunView .icon files
- images — for SunView .image files

language
Place a subdirectory in language for each language supported:

- USA-English
- English
- French
- French_Swiss
- German
- German_Swiss
- Italian
- Swedish
- Spanish

Four standard subdirectories are available for each language directory:

- doc — containing on-line documentation for the application in the particular language
- help — containing Spot Help and handbook files for the application in the particular language
- man — containing man pages for the application in the particular language
- messages — containing message files for the application in the particular language
Index

Symbols

application_name_ROOT variable 207, 145
& background character 126
.BAT files 111, 113
.COM files 111
.EXE files 111
.h files 200
.info files 81–86
.orgrc file
  description 68–73
  icons for 68
  parameters 69–72
  sample entries 72–73
  See also organizer program
.quick.pc file 114, 125
.rf files 87
.rgb file 105
.rgb(5) file format 255
/. See file system

Numerics

68000
  byte ordering used by 18
  byte swap problems 19
8-bit
  character handling 149
  displaying files in DOS Windows 117
  support 4, 149
80386 3, 17–19, 26–27, 33
80387 19
8086 33

A

a.out(5) file format 257

accents. See floating accent key
accounting cluster 142
adb(1) debugger 26, 255
  location 140
  manual describing 6
  Sun386i commands for 43
addresses, appearance of negative 30
adjacentscreens(1) command 257
advanced_admin cluster 142, 201
Alt Graph key 4, 22, 150, 152–153
Application SunOS 25, 28, 165
  contents 139–142
  manual describing 11
applications
  building and maintaining with the make(1)
    utility 6
  help, writing for 80–98B
  international 149–156, 161
  PC 3, 111, 160. See also MS-DOS
  porting
    C 247–250
    from UNIX System V 27
    from Sun-3 26–27
  releasing 144–146
  start_applic file 98A, 145–146, 204, 207
  subdirectories for 207–208
SunView, using 45
third-party software
  directories for 207, 144
  that run on Sun386i 4
  window concepts, manual describing 7
  window-based, creating 29
  See also color; graphics; MS-DOS;
  organizer(1) program
ar(1) command. See archiver (ar)
ar(4S) command 257
ar(5) file format 256
archiver (ar) 42, 255
/tmp directory, use of 199
as(1) command. See assembler (as)
asm keyword 40, 55
assembler (as) 25, 55, 33, 255
80386 versus 8086 33
expressions 173, 182
immediate values 182
input format 171–172
instruction descriptions 180–181
instructions
arithmetic/logical 185
control 182
conversion 186
coprocessor 186
decimal arithmetic 186
divide 186
tag 184
I/O 184
interrupt 187
jump 187
miscellaneous 188
multiply 186
new arithmetic 183
new bit 183
new condition code 182
new move 183
procedure call 187
processor extension 182–184
protection model 187–188
return 187
segment register 184
string 186–187
location 11, 143
mnemonics
addition 189
arithmetic 190
comparison instructions 190
constant instructions 190
division 189
integer transfers 189
multiplication 189
packed decimal 189
processor control instructions 191
real transfers 188
subtraction 189
transcendental instructions 190
object file, sections of 172
operands 178–179
operations
dbx pseudo 178
general pseudo 175–177
sdb pseudo 177
operators 173
output format 172
shared libraries, creating with 27
statements
assignment 171
empty 171
machine operation 171
modifying 172
pseudo operation 171
SunOS vs. Intel 80386 178, 188–191
symbols 173
syntax rules 173–175
test register 182
types 172–173
values 172–173
See also Common Object File Format (COFF)
AT bus 20, 159, 160
audit trail package cluster 142
auto.home(5) file 198, 254
auto.vol(5) file 206, 254
AUTOEXEC.BAT file 113
Automatic System Installation utility 98C–99
automounter (automount(8)) 195, 197–199, 206

B

bar(1) command 253
bar(5) file format 254
base_devel cluster 11, 143
bit flipping 19–20
bit shifting 33, 248
boards.pc file 113–115
boot
blocks, location 202
directory containing files for 199
servers, location 141
boot(8S) command 255, 257
Bourne shell
8-bit handling 149
manual describing 6
bus
AT 3, 20, 129, 159, 160
AT interface 131–135
comparison between Sun-3 and Sun386i 17
System 20, 159, 160
XT 3, 20, 129, 159
bwtwo(4S) command 256
byte ordering
affect on porting 18–19
byte swap problems 19–20, 59
comparison of 80386, VAX, and 680x0 18
graphics applications, correcting problem 45–46
network message passing 35
problems to avoid when porting C code 34–35
byte swapping 19–20, 59
byteorder(3N) function 256

C

26, 43
arithmetic conversions 240
assembler inlining (asm keyword) 40, 55
bit shifting, limit 33, 248
byte-ordering problems 19
cast operators 243
characters 239
compiler (cc) 25, 33, 255
–g and –go options 33, 223, 227–228
/tmp directory, use of 199
asm function declarations 40
assembly language use 171
casting a structure to a scalar value 39–40
line number information 219–220
location 11, 143
shared libraries, creating with 27
complex operations, replacing 53
data
alignment 38, 243, 247–248
layout 248
representations 36–37
sizes 247–248
types 247–248
declarations, inner and outer 242
documentation for 33, 6
double 239–240, 242, 248
evaluating conditions 53–54
float 239–240, 242, 248
functions 241–242, 249
generating string instructions 54
header files 200
improving loop efficiency 54–55
initialization 248
initializer 243
Kernighan and Ritchie C vs. Sun C 239–243
keywords
enum 239
void 239
asm 40, 55
linear code benefits 51–52
linkage rules 242
multiplicative operators 240
name spaces 239
optimizing code 48–55
pointers 241
portability and rules type checker for (lint) 6, 42, 249
porting problems, avoiding 33–36, 38–40
preprocessor (cpp), predefined symbols 39
primary expressions 240
registers 49–50, 248–249
right shifts 248
stack format 249–250
storage class specifiers 240
structure declarations 241, 243
switch expressions 242
type specifiers 240–241
unions 241, 243
variable names 247
C Programmer’s Guide for the Sun Workstation,
synopsis 6
calendar(1) program 141, 255
Catalyst program 4
cc. See C, compiler (cc)
CGA (Color Graphics Adapter) 109–110
CGI 26, 67
cgthree(4S) command 254
character codes 149–150
characters, creating West European 150–152, 4, 22
character sets
ISO 267–268, 154
ISO to MS-DOS conversion 269–271
MS-DOS 262–263
MS-DOS to ISO conversion 264–266
chess(6) command 257
chesstool(6) command 257
chown(8) system call 30
click(1) command 255
client(8) command 255
cluster(1) command 12, 143, 253
clusters
accounting 142
advanced_admin 142, 202
Application SunOS 141–142
audit 142
base_devel 11, 143
comm 142
config 11, 143
database for 201
definition 11
determining if loaded 143
Index Continued

<table>
<thead>
<tr>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>disk_quotas</td>
<td>142</td>
</tr>
<tr>
<td>doc_prep</td>
<td>142</td>
</tr>
<tr>
<td>extended_commands</td>
<td>142</td>
</tr>
<tr>
<td>games</td>
<td>142, 201</td>
</tr>
<tr>
<td>help_guide</td>
<td>11, 79, 91, 143</td>
</tr>
<tr>
<td>loading</td>
<td>12-13</td>
</tr>
<tr>
<td>location if loaded</td>
<td>200</td>
</tr>
<tr>
<td>mail_plus</td>
<td>142</td>
</tr>
<tr>
<td>man_pages</td>
<td>142</td>
</tr>
<tr>
<td>name_server</td>
<td>142</td>
</tr>
<tr>
<td>networking_plus</td>
<td>142</td>
</tr>
<tr>
<td>old_commands</td>
<td>142</td>
</tr>
<tr>
<td>plot</td>
<td>142</td>
</tr>
<tr>
<td>plot_devel</td>
<td>11, 143</td>
</tr>
<tr>
<td>proflibs</td>
<td>11, 143</td>
</tr>
<tr>
<td>sccs</td>
<td>202</td>
</tr>
<tr>
<td>size of, displaying</td>
<td>143</td>
</tr>
<tr>
<td>spellcheck</td>
<td>142, 201</td>
</tr>
<tr>
<td>SunOS Developer's Toolkit</td>
<td>11</td>
</tr>
<tr>
<td>sunview_devel</td>
<td>11, 143</td>
</tr>
<tr>
<td>sysV_commands</td>
<td>142</td>
</tr>
<tr>
<td>sysV_devel</td>
<td>12, 143</td>
</tr>
<tr>
<td>unloading</td>
<td>12-13</td>
</tr>
<tr>
<td>viewing information about</td>
<td>12</td>
</tr>
</tbody>
</table>

**Code**
- control. *See Source Code Control System (sccs)*
- optimization 48-55
- position-independent (PIC) 49

**COFF (5) file format** 254
- COFF. *See Common Object File Format (COFF)*

**Color**
- applications 5, 66, 99-105, 166
- colormap definition 99
- emulation 109-110
- foreground and background 100-101
- guidelines for using 105-106
- panels, adding 101-104
- *See also coloredit(1) program*

**Color Graphics Adapter (CGA) emulation** 109-110
- coloredit(1) program 26, 66, 104-105, 166, 253
- on-screen documentation for 77
- colormap, definition 99
- comm cluster 142
- command interpreter (/sbin/sh) 199

**Commands**
- dos2unix(1) 261
- Sun386i
  - altered for 255-257
  - new 253-255
  - not pertaining to 257
- unix2dos(1) 261

*See also* individual command name; *man* pages

**Common Object File Format (COFF)** 25-26, 28, 31
- a.out header 214-216
- addresses, physical and virtual 213
- auxiliary symbol table entries 231-235
  - C structure declaration for 234-235
  - for arrays 233
  - for beginning of blocks and functions 233
  - for end of blocks and functions 233-234
  - for end of structures 232
  - for enumeration symbols 234
  - for filenames 232
  - for functions 232-233
  - for sections 232
  - for structure symbols 234
  - for tag names 232
  - for union symbols 234
- external symbol representation in 33
- features 211
- file header
  - contents 213-214
  - optional information 214-216
- functions, symbols for 222
- line number information 219-220
- link editor SECTIONS directive, use of 218
- magic number 213, 215
- man pages for 253-254
- object file sections 211-212
- reading parts of files with access routines 236
- reloc.h file 219
- relocation information 218-219
- section header
  - .bss section 218
  - C structure declaration 217
  - flags 216-217
  - table 216
- section numbers 226-227
- sections, description 212-213
- storage classes 224-227, 229-230
- storclass.h file 223-224
- string table 235-236
- symbol table entries 222-231
- System V functions for manipulating 32
- type entries
  - by storage class 229-230
  - derived types 228
  - fundamental types 228-229

**communications. See networks**
**compilers. See C compiler (cc); yacc compiler**
Compose key 4, 22, 150–152
Computer Graphics Interface. See SunCGI
config cluster 11, 143
config(8) file format 256
CONFIG.SYS file 113
configuration files, location 141
core system 140–141, 165
core(5) file format 256
cpp(l) command 255, 39
CPU board, contents 163
creat(2) system call 30
csh(l) command 255
Curses facility, manual describing 6

d
D

daemons, location 142
database software 55
dbx(l) debugger 26, 255
C code, use with 33
location 11, 143
manual describing 6
object file information 223
Sun386i registers for 44–45
debuggers. See adb(l) debugger; dbx(l) debugger; kadb(8S) debugger
DEFINE_ICON_FROM_IMAGE macro 45–46
Developer’s Toolkit. See SunOS Developer’s Toolkit
device drivers 129–131
manual describing 7
loadable 129
relationship to Pixrect graphics library 20
timing dependencies 130
See also MS-DOS, drive designations
devices
adding 19, 129–131
directory for 198
mass storage 162
See also device drivers
diagnostics 139–140, 164, 199
directories
exporting 96–98, 204
graphically displaying with organizer 68
See also file system
disassembler (dis) 42, 253
diskless systems, files for 199
disk_quotas cluster 142
disks
capacity of for Sun386i 20
quot(8) command, location 142
dkio(4S) command 256
doc_prep cluster 142
documentation
Developer’s Toolkit Documentation Set 6–7, 168
for C 6, 33
for debuggers 6
for FORTRAN 41
for network programming 6
for Pascal 41
for Pixrect graphics library 7
for programming utilities 6
for PROM, ID PROM, and EEPROM 6
for Sun386i, synopsis 167–168
for SunCGI 7
for SunOS, operating system 6
for SunView 7
for writing device drivers 7
on-screen topics 77, 88
Owner’s Set 168
Owner’s Supplement Documentation Set 168
Upgrade Documentation Set 168
dos(l) program
& background character 126
.BAT files 111
.COM files 111
.EXE files 111
.quick.prm file 114, 125
8-bit handling 117, 149
boards.prm file 113–115
code set 150
description 109–110, 153
DOS_CMDTOOL environment variable 125
DOSLOOKUP environment variable 112, 117, 125
drive C:, space issue 124
EDITDOS program, source code for 118–123
expanded memory for applications 111
I/O address space emulation 115
LIM memory use 111
name of process running in, ensuring unique 117
on-screen documentation for 77
opening implicitly 110
piping 125–126
port limitations 123
scratch files, conflicts with 124
screen height limitations 123
setup.prm file 112–115
SunOS commands, invoking from 111, 125
text-only applications, running 112
See also MS-DOS
DOS Windows. See dos(1) program
dos2unix(1) command 116–117, 150, 253, 261
DOS_CMDTOOL environment variable 125
DOSLOOKUP environment variable 112, 117, 125
drivers. See device drivers
dump(8) command 256

E
ed(1) editor 141, 199
editors
  crash files for ed and vi 199
  location 141
  See also link editor (1d)
EDITDOS program, source code for 118–123
EEPROM, manual describing 6
EGA (Extended Graphics Adapter) 110
encryption, files and keys for 141
environment variables
  DOS_CMDTOOL 125
  DOSLOOKUP 125
  $application_name_ROOT 207, 145
  $PATH 145–146, 201
environmental requirements 163
error messages, rewording from kernel 73–76
Ethernet 26, 55
expansion unit 19, 160
exporting directories, steps for 97–98
Extended Graphics Adapter (EGA) 110
eXternal Data Representation (XDR) 26, 55
  ensuring correct network byte order with 35
  using as data format 47

F
f77(1). See FORTRAN
fbio(4S) command 256
fcntl(2) system call 30
fd(4S) command 254
file conversion
  between MS-DOS and 8-bit text 117
  between SunOS and MS-DOS text files 116–117
file system
  (/root) 196–200
  /bin 197
  /boot 197
  /dev 198
  /etc 198
  /export 197, 204–205
  /files 197, 203–204
  /home 196–198
  /kadb 198
  /lib 198, 201
  /lost+found 198
  /mnt 198
  /net 198
  /sbin 199
  /sys 199
  /tftpboot 199
  /tmp 199
  /tmp_mnt 199
  /usr 196–197, 199–203
  /var 199
  /VERSION 200
  /vmunix 200
  /vol 200, 206–207
applications, subdirectories for 207–208
Berkeley UNIX commands (/usr/ucb) 202
boot directory (/usr/boot) 200
C header files (/usr/include) 201
checking and repairing (with /sbin/fsck) 199
clusters
  database for (/usr/lib/load) 201
  location if loaded (/usr/cluster) 200
command interpreter (/sbin/sh) 199
configuring network (with
  /sbin/netconfig) 199
diagnostics, files for (/var/sysex) 199
home directory 197–198
kernel core dumps, location (/var/dump) 203
mounting (/sbin/mount) 199
network configuration file
  (/sbin/netconfig) 199
printing files (/var/spool) 199
process control initialization (/sbin/init) 199
rebooting, file for (/sbin/reboot) 199
root file system (/) 196–200
spell program database (/usr/dict) 201
Sun386i, overview 195–197
system administration directories (/etc and
  /usr/etc) 198, 201
temporary files, mount points for
  (/tmp and /tmp_mnt) 199
UNIX System V
  binaries (/usr/5bin) 200
  include files (/usr/5include) 200
  libraries (/usr/5lib) 200
utilities directory (/usr/lib) 201
VERSION file 200
See also Application SunOS; clusters; SunOS
Developer’s Toolkit
filesizes file 140, 142–143, 201
floating accent key 150, 154
fonts
converting between 150
pcfont .b .14 149
pcfont .r .14 149
screen . iso . r .12 149
FORTRAN 26, 40–41, 43, 165
frame buffers
byte swap and bit flip problems 21
size, affect on porting 20
Sun-3 vs. Sun386i systems 17
Sun386i 20, 164
functions, for manipulating COFF files 32

G

GKS (Graphics Kernel System) 26, 67
gprof(1) command, description 42
graphics
bit-flipping problems 19
byte-order problem, avoiding 45–46
Color Graphics Adapter (CGA) emulation 110
device-independent, using RasterOP library
routines to create 7
Extended Graphics Adapter (EGA) card, adding 110
Hercules emulation 110
icons, creating for application files 68–73
interface to SunOS file system 68–73
libraries for 11, 26, 67
Monochrome Display Adapter (MDA) emulation 110
packages for 25
pixrect suggestions 46
porting 65
software for 165–166
standard word format for 21
Sun-3 vs. Sun386i systems 17
SunCGI, use for 7
SunView and Pixrect libraries, use for 7, 20
See also Pixrect graphics library; SunView
Graphics Kernel System (GKS) 26, 67
group(5) command 254

H

handbooks. See Help Viewer
hardware
diagnostics 139–140
porting issues 17–22
Sun-3 vs. Sun386i system 17
help. See on-screen help
Help key 4, 22
Help Viewer
command for 253
description 66, 76–79
file format for 254
guidelines for 94
handbooks
appearance, checking 95
making user visible 95–98B
templates for 93–94
topics, referring to 85–86
writing for applications 86–95
Help Writer’s Handbook, loading 88
Master Index 79
on-screen documentation for 77
sample window 78
table of contents 79
See also on-screen help
help(5) file format 254
HELP_DATA attribute 81–85
help_guide cluster 11, 79, 91, 143
help_guide directory 87
help_viewer(1) command 253
help_viewer(5) file format 254
Hercules Graphics Adaptor emulation 109–110
home directory 197–198, 203
hypertext links
definition 78
description 88–89
implementing 89–93

I

I/O ports
Ethernet 19
parallel 19
RS-423 19
SCSI 19
Sun-3 vs. Sun386i systems 17
IBM 360, byte ordering used by 18
iconedit(1) 45–46
icons, creating for applications 68–73
ID PROM, manual describing 6
ie(4S) command 256
include files, containing COFF definitions 31
inet(3N) function 256
inet(4F) protocol family 256
Intel 3
interfaces 19
internat(5) file format 254
international applications 149–156, 4
International Standards Organization. See ISO
interrupt channels 116
ipalloc(3R) command 254
ipalloc.netrange(5) file format 254
ipallocd(8C) command 255
ISO character set 150, 267–271

K
kadb(8) debugger 26, 131, 198, 257
kernel
  8-bit handling 149
core dumps, location 199
debugger for, location 198
location 140, 200
messages, translating 73–76, 154
reconfiguring, files for 202
keyboard 161
  compatability 4
description and illustration of Sun386i (U.S.
and Great Britain) 22
layout 161
Sun-3 vs. Sun386i system 17
keys
  Alt Graph 4, 22, 150, 152–153
  AT-style 4
  Compose 4, 22, 150–152
  floating accent 150, 154
  Help 4, 22
  Sun-3 4, 161
  West European characters, creating 150–152,
  4, 22
 keystation map
  international 153
  -U.S. 152
kill(l) system call 30
kill(2V) system call 30

L
ld(l) command 255
ld* files 32, 254
lex(l) lexical analysis program, manual
describing 6
libld.a library 236
libraries
  for graphics applications 26, 67
  libld.a 32, 236
  listing of Sun386i 28
  lorder(l) utility for 42
  plotting, location 11, 143
  profiled, location 11, 143
  ranlib(l) utility for 42
  SunView, location 11, 143
  See also archiver (ar); profiled libraries;
  shared libraries
lightweight processes 27
LIM (Lotus-Intel-Microsoft) memory 111
link editor (ld) 42
  a.out header 214–216
  C linkage rules 242
  location 11, 143
  relocation information, use of 218–219
  SECTIONS directives 218
  shared libraries, creating with 27–28
  See also Common Object File Format
  (COFF)
links
  between MS-DOS and SunOS files 109
  hypertext 78, 88–93
lint(lV) program checker 42, 6
load(l) command 12, 141, 143, 253
loadable drivers 129
loadc(l) command 12, 143, 253
logintool(8) command 255
lorder(l), on Sun386i 42–43

M
m4(lV) macro processor, manual describing 6
machid(l) command 255
macros
  DEFINE_ICON_FROM_IMAGE 45–46
  processor (m4) 6
mail(l) program
  location 141–142
  on-screen documentation for 77
mail_plus cluster 142
make(l) command
  building and maintaining programs with 6
  running on MS-DOS targets 112
man pages
  COFF-related 31
dis(1) 253
ldahread(3X) 254
ldclose(3X) 254
ldfhread(3X) 254
ldgetname(3X) 254
ldlread(3X) 254
ldlseek(3X) 254
ldohseek(3X) 254
ldopen(3X) 254
ldrseek(3X) 254
ldshread(3X) 254
ldssseek(3X) 254
ldtbindx(3X) 254
ldtbread(3X) 254
ldtbseek(3X) 254
objdump(l) 253
location on line 142
Sun386i
altered for 255–257
new for 253–255
not pertaining to 257
See also individual command name
man_pages cluster 142
markers, for hypertext links 88–93
mass storage devices 162
math coprocessor 19
mc68881version(8) command 257
MDA (Monochrome Display Adapter) 109–110
mem(4S) command 256
memory
main 22, 17
Sun-3 vs. Sun386i systems 17
messages, translating 154
mknod(2, 8) command 30, 126
modload(8) command 255
modstat(8) command 255
modunload(8) command 255
monitor(3) function 256
monitor(8S) command 257
monitors
compatibility with existing 161
sizes for Sun386i 22
Sun-3 vs. Sun386i systems 17
Monochrome Display Adapter (MDA) emulation
109–110
mouse 17, 161
mpr_static routine 45–46
MS-DOS 109–110, 3, 25–28, 160, 166
& background character use 126
adding devices to run under 129–130
applications, naming 111
AUTOEXEC.BAT file 113
CGA emulation 26
character set for 262–263
character conversion, to ISO characters 264–266
code set 150
command line interpretation 117
commands
invoking from SunOS prompt 125
invoking SunOS commands at MS-DOS prompt 125
CONFIG.SYS file 113
drive designations 113
EGA support 26
Hercules emulation 26
international fonts for 149
limitations 123–124
location 141
make files, running on DOS targets 112
MDA emulation 26
piping 117, 124–126
text file conversion 116–117, 150, 264–266
See also dos(1) program; DOS Windows

N
name_server cluster 142
netconfig(8C) command 255
Network File System (NFS) 55, 3, 26, 195
files for, location 141
manual describing 6
Network Programming on the Sun Workstation, synopsis 6
networking_plus cluster 142
networks
/vol directory 206–207
byte-ordering problems 35
commands, location 142
configuration file for (/sbin/netconfig) 199
DOS Windows number, using 117
exporting directories and files 203
file system considerations 195
files for, location 141
manual describing 6
native executables for servers 203
root directories for clients 204
software for 55
swap directories for diskless clients 204
Yellow Pages 195
Index  Continued

See also automounter (automount (8)); file system
New User Accounts utility  98C–99
NFS. See Network File System
nlist(3) function  256
rm(1) command  42, 255
nroff(1) program, location  142

O
objdump(1) command  253
object code
disassembler (dis)  42
dumper (objdump)  42
produced by C, FORTRAN, and Pascal  43
object files
access routines for reading parts of  236
displaying call-graph profile data  42
format  172
functions for manipulating COFF  32
printing name list of  42
printing section sizes of  42
removing symbol and line number information from  42
tools for  31
See also Common Object File Format (COFF); assembler (as)
on-line help. See on-screen help
on-line error messages, rewording  73–76
on-screen help
.info files  81–86
description  66, 76–79
Done button  77
handbooks for  77
Help Viewer description  66, 76–79
Help Writer’s Handbook, loading  88
HELP_DATA attribute  81–85
hypertext links  78, 88–93
location of files  88, 141
Master Index  79
More Help button  77
Spot Help description  76–77, 66
Sun386i topics, referring to  85–86
sun_external.info file  85–86
table of contents  79
Top Level link  78
writing
guidelines  86, 94, 11, 143
Help Viewer text for applications  86–95
Spot Help text for applications  80–86
templates for  93–94
open(2V) system call  30
operating system, program that loads  197. See also SunOS operating system; MS-DOS operating system
optimization
assembler inlining  55
complex operations, replacing  53
conditions, evaluating  53–54
linear code benefits  51–52
loop efficiency, improving  54–55
methods  48–55
register use to enhance  48–51
string instructions, generating  54
optional clusters  139, 165
contents of  141–142
listing loaded  12
loading and unloading  12–13
See also SunOS Developer’s Toolkit
organizer(1) program  26, 253
.orgrc file  68–73
on-screen documentation for  77

P
panels, adding color to  101–104
Pascal  41, 43
passwd(1) command  255
PC applications
namings  111
scratch files, conflicts with  124
SunOS commands for  111
See also MS-DOS; dos(1) program
pcfont.b.14  110, 149
pcfont.r.14  110, 149
performance
analysis, manual describing  6
impact of virtual memory management on  27
peripheral expansion  19
PIC  48
piping, between SunOS and MS-DOS  117, 124–126
Pixrect graphics library  7, 20, 26, 45, 65
plot cluster  11, 142
plot_devel cluster  11, 143
plotting, location of files and libraries for  11, 142–143
pnp(3R) command  254
pnpboot(8C) command  255
pnpd(8C) command  255
policies(5) file format  255
porting
byte-ordering issues affecting  18
C code 33–36, 38–40
checklist 62
frame buffer size issues 20
graphics applications 21, 65
hardware overview 17
large programs 30
non-UNIX applications 30
processor issues affecting 18
screen resolution issues 20
software overview 25–26
standard data format for 47–48
summary 59–62
Sun-3 applications 26–27
tracing system calls 43
UNIX-based applications 27, 29
position-independent code (PIC) 48
power supply 160
pp(4) command 254
pr_flip routine, description 45–46
prntcap(5) command 256
printing, location of files for 141, 203
processors, Sun-3 vs. Sun386i systems 17
prof(1) command, description 42
profiled libraries, listing of Sun386i 28
proflibs cluster 11, 143
Program Debugging Tools for the Sun Workstation, synopsis 6
programming utilities, manual describing 6
PROM, manual describing 6
protocols, manual describing 6
ptrace(2) command 256

Q
quot(8) command, location 142

R
ranlib(1) command 42–43
rarpd(8C) command 257
RasterOP graphics library
   manual describing 7
   RasterOp routines 20, 65
rc(8) command 256
rebooting, file for (/sbin/reboot) 199
reloc.h file 219
Remote Procedure Call (RPC)
   administration facilities use of 98C
   description 55, 26
   file format used by 47
   files for, location 141
   manual describing 6
roffbib(1) command 255
root file system (/) 196–200
root(4S) command 254
RPC. See Remote Procedure Calls

S
sccs cluster 202
scratch files, avoiding conflicts with 124
screen.iso.r.12 font 149
screens
   international fonts for 149
   resolution, affect of on porting 20
   updating facility for (Curses) 6
SCSI controller 129, 160
sd(4S) command 256
setkeys(1) command 152, 255
setup.pc file 112–115
sgetl, description 32
shared libraries 27–28
Show Map, organizer feature 68
SIMM board 22, 164
Single In-line Memory Module (SIMM) 22, 164
size(1) command 42, 255
snap(1) program 26, 98C, 141, 253
software
   database 55
   development tools 31
   network 55
   Source Code Control System (sccs) 6, 12, 143
   Sun-3 vs. Sun386i systems 25
   Sun386i, summary of 25
   See also applications; file system; porting;
   system software
SPARC, byte ordering used by 18
spell(1) program and databases 142, 201
spellcheck cluster 142, 201
Spot Help
   description 66, 76–77
   on-screen documentation for 77
   writing
      for applications 80–86
      guidelines for 86
   See also on-screen help
sputl, description 32
st(4S) command 256
start_applic file 202
storclass.h file 223–224
streams applications programming, manual describing 6
strip(1) command, description 42
Sun System Services Overview, synopsis 6
Sun-3 system
  hardware comprising 17
  porting applications from 26–27
  software comprising 26
Sun386i system
  application development goals 4
  character codes supported 149
  configurations 4
  CPU board, contents 163
  diagnostics 164
  dimensions 162
  disk capacity 20
  DMA channel assignments 132
  documentation 5–6, 167–168
  environmental requirements 163
  expansion unit 160
  file system layout, overview 195–197
  frame buffers 164
  graphics software 165–166
  hardware summary 17
  I/O ports for 19
  interrupt channels 132
  interrupt level availability 116
  keyboard 161
  languages supported 165
  lightweight process capability 27
  monitors 161
  mouse 161
  MS-DOS 166
  object file tools 31
  on-line message categories 73
  overview 3
  power supply 160
  software summary 25
  system administration features 4, 98C–99, 167
  unbundled software 166
  user interface features 167
  weights 162
  window-based applications 66, 165–166
See also snap(1) program; coloredit(1) program; dos(1) program; on-screen help; organizer(1) program
SunCGI  7, 26, 67, 166
SunGKS  26, 67, 166
suninstall(8) command 256
SunOS Developer's Toolkit  25, 28, 139, 165
  clusters 12, 143
SunOS operating system
  changes between 3.x and 4.0 27
  commands, location 200
  description 26, 3
  devices, adding to run under 130
  kernel, file containing (/vmunix) 200
  loading, program for 197
  manual describing 6
  overview 27–28
  version of, determining 200
SunSimplify  55, 26, 166
suntools  26
suntools_devel cluster 11, 143
SunUnify  55, 26, 166
SunView  3–4, 26, 165
  1.75 enhancements 66
  color basics 99–101
  development libraries, location 11, 143
  graphics applications, use for 20
  location 141
  manual describing 7
  on-screen documentation for 77
  tools 25
swap directories, for diskless clients 204
switcher(1) command 257
symbol tables, produced by C, FORTRAN, and Pascal 43. See also Common Object File Format (COFF)
symorder(1) command 255
syscall(2) command 256
sysex(1) command 253
syscall(3) command 73
syslog.conf file 74, 76
syslog.conf(5) file format 256
syslogd(8) command 256
syslogd(8) daemon 73
system administration
  directories for 198, 201
  features 98C–99, 167
  files for, location 142, 200–201, 203
System bus 17, 159, 160
system calls
  chown(8) 30
  creat(2) 30
  fcntl(2) 30
  kill(1) 30
  kill(2V) 30
  mknod(8) 30
open(2V) 30
utime(3C) 30
System Exerciser 164, 199
system interfaces 19
system software
  accounting files 142
  adb(l) debugger 140
  snap(l) files 141
  Application SunOS 25, 139–141
  assembler (as) 11, 143
  audit package 142
  boot servers 141
  C compiler (cc) 11, 143
  calendar(l) program 141
  configuration files 141
  core system 140–141
  daemons 142
dbx(l) debugger 11, 143
encryption files 141
games files 142
libraries 11, 143
link editor (ld) 11, 143
mail files 141–142
man pages 142
MS-DOS 141
network commands 142
network files 141
on-screen help files 141
optional
  description 141–143
  loading 12–13
plotting files and libraries 11, 142–143
printing commands 141
quot(8) command 142
root file system (/) 140
Source Code Control System (sccs) files 12, 143
spell(l) program 142
SunView 141
Hardware Diagnostics 139–140
tip(lC) command 142
uucp(lC) command 142
UNIX System V commands 142
See also Application SunOS; clusters; file
system; SunOS Developer’s Toolkit
system unit, contents 159–160
sysV_commands cluster 142
sysV_devel cluster 12, 143
syswait(l) command 253

T
tcov(l) command 257
temporary files, location 199
textedit(l) program
  8-bit handling 149
  on-screen documentation for 77
tftp(lC) command 257
tip(lC) command, location 142
toc(5) file format 255
Top_Level file 79, 87, 95–96
trace(l) command, description 43
translate(5) file format 255
troff(l) program, location 142

U
unconfigure(8) command 255
UNIX System V 3, 27, 31
  binaries for 200
  COFF files, functions for manipulating 32
  COFF format 211–236
  commands, location 142
  compatibility with SunOS operating system 6, 29–30
  include files for 200
  kernel, files needed to reconfigure 11, 143
  libraries 200
  porting from 12, 27, 143
  STREAMS interface 6
  tools 26
unix2dos(l) command 116–117, 150, 253, 261
unload(l) command 12, 13, 143, 253
unloadc(l) command 12, 13, 143, 253
unshared libraries, specifying 28
utime(3C) system call 30
uucp(lC) command, location 142

V
VAX, byte ordering used by 18
vgrind(l) command 257
vi(l) editor 141, 199
volumes 206

W
windows
  applications that run in 66
  on-screen help for, creating 80
  developing applications that run in 7
Index Continued

divisions of system for 65–67
foreground and background colors for 100–101
software for 165–166
tools for 25
See also DOS Windows
Writing Device Drivers for the Sun Workstation,
synopsis 7

X

XT bus 20, 159

Y

yacc(1) compiler compiler, manual describing 6
Yellow Pages (YP) database 55, 26, 195
files, location 141
manual describing 6
use of 98C
YP. See Yellow Pages database
ypupdated(8C) command 257

Z

zs(4S) command 256