Release Notice:

DG/UX for AViiON Systems Release 4.30

July 1990

Part Number 085-600125-03

This Release Notice applies to Models:
Q001A P001A

DATA GENERAL CORPORATION
4400 Computer Drive
Westboro, Massachusetts 01580

CONTENTS

1. Introduction

This Release Notice describes the DG/UX™ System product for Data General's family of AViiON™ computers. In addition, this notice includes information not currently available in the product manuals (e.g., information developed after the current manuals were printed, or corrections to current manuals).

You may load release 4.30 of the DG/UX system on a clean (empty) root and **/usr** file system, or you may choose to overload a DG/UX Release 4.20 (or higher) system. See chapter two of the 4.30 revision of *Installing and Managing the DG/UX System* for more information on software loading procedures.

-- WARNING --

There may be certain layered products that depend upon this release of DG/UX or that might not work correctly with it. Please contact your Data General representative for more information.

You may print additional copies of this release notice after you have installed the DG/UX System. A copy suitable for lineprinters can be found in the file **/usr/release/dgux_4.30.rn**. In the event of (actual text) differences between the printed copy of this notice and the online lp version, the printed copy takes precedence.

2. Product Description

DG/UX 4.30 is the standard operating system for Data General's line of industry-standard systems based on the Motorola 88100 Reduced Instruction Set Computer (RISC) microprocessor. DG/UX offers AViiON family support for workstations, servers, uniprocessor and N-way fully symmetric multiprocessors.

DG/UX for AViiON systems is an advanced implementation of the UNIX® operating system based on AT&T's System V.3 and the Berkeley Software Distribution.

The DG/UX software may be bundled on the same tape with other products. In any case, DG/UX software components consist of:

- DG/UX kernel, commands and libraries.
- GNU C compiler
- DTK: The Documenter's Tool Kit
- DG/UX on line Man Pages

The following is a list of standards-based features of DG/UX 4.30:

- System V.3 functionality including STREAMS as defined by SVID Issue 2.
- All popular BSD features including socket-based TCP/IP compatibility.
- POSIX.1 conformance (IEEE Standard 1003.1-1988).
- Certified compliance to 88open BCS Release 1.0, including support for both generating and executing BCS-conforming programs.
- SVID Issue 2 conformance, verified with SVVS.
- ANSI-compliant C compiler optimized for 88000/RISC systems.
- ONC-NFS™ 4.0 compatibility.
- IEEE floating-point conformance (IEEE Standard 754-1985).
- X Windows Version 11 Release 3.0 compatibility with OSF/Motif interface.

- SunOS 4.0 tape format compatibility for installation, operation and use of diskless clients and servers in mixed SunOS and DG/UX environments.

3. Environment

3.1 Hardware

Release 4.30 of the DG/UX System will run on Data General AViiON series machines with revision E.2 or later of the MC88100 processor and a minimum of 8-megabytes of main memory, at least 179-megabytes of disk storage, a system console (or graphics monitor for workstations), and a tape drive (for software distribution). Consult your sales representative for complete information on disk drives, tape drives, printers, and terminals supported by the DG/UX System.

Release 4.30 of the DG/UX System may also be run on a "diskless" AViiON workstation. This requires a minimum of 8-megabytes of main memory and console or workstation monitor, but does not require any disk or tape drive units. If you plan to be doing work that puts a very heavy windowing load on the system, we suggest you consider using at least 12-megabytes of main memory.

Previous releases of DG/UX do not support SCSI synchronous transfers on the AV5000/AV6000 Series of servers due to a controller firmware limitation. Revision 11 firmware for the VME SCSI controller removes this limitation. During system boot of DG/UX , the controller revision is checked and if it is not Revision 11 or later, the message:

```
Firmware in SCSI controller "cisc()" is out of date --
see release notice.
```

is printed on the system console. This message indicates that the current controller firmware will not support synchronous transfers. There is no risk in using firmware revisions prior to revision 11 since asynchronous SCSI transfers will continue to work. See Section 5.3 "Kernel - Synchronous SCSI protocol" for more information on this message.

4. Enhancements and Changes

This section describes enhancements and changes for Release 4.30 of the DG/UX System.

4.1 General

BCS Certification

The DG/UX Release 4.30 execution environment has been certified compliant to the 88open Binary Compatibility Standard (BCS), Release 1.0. BCS-certified applications will run on DG/UX Release 4.30 without change. Also, DG/UX Release 4.30 has been certified as a compliant generation platform. BCS compliant applications can be developed on DG/UX Release 4.30.

Documentation

The *Installing and Managing the DG/UX System* manual has been upgraded for release 4.30. In particular, chapter 2 now contains all of the information that was in section 8 of the 4.20 DG/UX release notice.

4.2 Kernel

Swap space

Beginning with DG/UX Release 4.30, user processes are charged for swap space as they allocate virtual memory (e.g., with **malloc, brk**, or **fork**) rather than when the swap space is actually used. This prevents overcommitment of the available swap space. However, programs which ran successfully under previous DG/UX revisions may now fail attempting to allocate virtual memory. If this occurs,

```
out of paging area space
```

diagnostics will appear on the system console. In this event, correct any programs which are known to allocate virtual memory that is never used. Alternatively, allocate more swap space for your system. For servers and work stations with local swap space, this can be accomplished **without** rebooting. Create another LDU with diskman but don't make a file system on it. Use the **swapon** command to put it into service. See **swapon(1m)** for details on adding it to the appropriate scripts and files for when you **do** reboot in the future. For diskless work stations, your swap file (/srv/swap/*host*) will have to be enlarged and your system be rebooted for it to take effect. Use the following command to make a 50,000 block swap file:

```
/usr/admin/bin/mkfile 25165824 /srv/swap/host
```

Floating Point

- Several minor bugs in the handling of infinities, NaNs, and floating point exceptions have been fixed.
- Floating point arithmetic on denormalized numbers is now fully supported.

SMD disk controller

A kernel fix is supplied which rectifies a problem in which the SMD disk controller was returning error code 0x23.

Ciprico SCSI host adapter

The Ciprico SCSI host adapter driver, **cisc**, has been enhanced to return more than 8 bytes of sense data when a SCSI operation completes with a "check condition" status.

New devices

Support has been added for several types of optical mass storage devices:

- a CD-ROM drive supporting 600-megabyte removable disks
- a Magneto-Optical drive supporting 590-megabyte and 650-megabyte removable disks

CD-ROM formats

The High Sierra, ISO 9660, and DG/UX physical disk format on CD-ROM are being supported for access both locally and via NFS.

Synchronous SCSI protocol

Synchronous SCSI protocol is supported on the AV5000/AV6000 series. Refer to the **Kernel** section of *Notes and Warnings* in this Release Notice for further details on this new feature.

Concurrent page out

The maximum number of concurrent page out operations has increased from one to six for all systems except diskless systems.

Message Queues

Message Queues performance has been enhanced.

Semaphores

Semaphore performance has been enhanced.

SYAC device

The Asynchronous Terminal Controller code has been upgraded.

Memory-based file systems

Support has been added for memory-based file systems. See **mfs(4)**. Also observe the following warnings when using memory file systems:

- Be careful not to overcommit the swap space available to the system. Because of the way DG/UX allocates memory, if you create a large memory file system, run some large application, then fill the memory file system, you may exhaust the swap space on the system. This will cause the system to kill arbitrary processes in order to recover the swap space.
- Do not use the use_wired_memory option on diskless work stations; it can cause system failures.

Kernel Debugger

We are providing *run-time* kernel debugging capability in DG/UX release 4.30. See *Using the DG/UX System Kernel Debugger* manual for details on building kernels with this internal debugger.

4.3 Header files

limits.h

Support for the System V.3-based limits macros from **<limits.h>** (e.g. ARG_MAX, PATH_MAX, OPEN_MAX, etc.) is now available by defining the preprocessor macro _SYSV3_LIMITS_FLAVOR. These macros were unavailable in DG/UX Release 4.20 since they pertain to runtime configurable parameters that shouldn't be hard coded.

Use of these macros is discouraged, in favor of the values returned by **sysconf, pathconf,** and **fpathconf**.

locale.h

In order to move closer to conformance with the 88open OCS, **<locale.h>** has a new category, LC_MESSAGES. This change resulted in the value of LC_ALL going from 5 to 6.

math.h

The prototype for **atof** in **<math.h>** was corrected to agree with the prototype in **<stdlib.h>**.

regexp.h

The header file **<regexp.h>** now has function prototypes and has been changed so that most ANSI C compiler complaints have been fixed.

stdio.h

The header file **<stdio.h>** now has function prototypes for **__filbuf** and **__flsbuf** in order to eliminate compiler complaints.

sys/ioctl.h

The following **ioctl** commands have been given new values for compliance with 88open standards:

- FIOGETOWN
- FIOSETOWN
- FIOASYNC
- FIONREAD
- FIONBIO

Existing device drivers which recognize any of these commands should be changed to recognize both the new and old values. The old values have been renamed using the prefix "_DG_OLD_".

termio.h and termios.h

The header **<termio.h>** has been changed for compliance with 88open standards. Executable programs compiled using older versions of the **<termio.h>** header will continue to work as before; no recompile is required. Source compatibility is affected in the following ways:

- The special characters VSTART, VSTOP, and VSUSP are no longer supported by struct termio. Any program which requires the ability to access the values of these special characters must be changed to access them through the POSIX terminal interfaces supported by **<termios.h>**.
- The headers **<termio.h>** and **<termios.h>** are no longer compatible for inclusion together within a single source program. Any program including both headers should be changed to include only one. A program needing features from both headers may include both by defining either _POSIX_BAUD_RATE_FLAVOR or _SYSV3_BAUD_RATE_FLAVOR before including the system headers. The POSIX and struct termio baud rate constants are incompatible; only one of these two sets may be used, and only with its designated interfaces (i.e., POSIX or termio).
- The flags members of struct termio have changed in length. Programs that use pointers to these member variables may require changes to the pointer data types.

- Most of the contents of **<sys/termio.h>** are now unsuitable for use within device driver source code. This is because the **ioctl** commands now defined within this header are translated outside the kernel to BCS **ioctl** commands. Any device driver which has been supporting commands defined within this header should be changed to recognize the corresponding BCS commands (defined within **<sys/_int_posix_tty_ioctl.h>**) instead.

time.h

The definition of the macro CLK_TCK in **<time.h>** has been modified so that inclusion of **<unistd.h>** is no longer necessary.

utmp.h

In the header file **<utmp.h>**, the macro **nonuser** has been fixed to eliminate LOGIN processes as nonusers.

varargs.h and stdarg.h

Both **<varargs.h>** and **<stdarg.h>** have new Green Hills sections to eliminate compiler warnings. These two include files provide support for both Green Hills **and** the GNU 'C' compilers.

4.4 System Calls

adjtime

The Berkeley **adjtime** system call has been implemented. See **adjtime(2)** man page for more information.

dg_xtrace

Dg_xtrace has been enhanced by the addition of stop-on-store capability.

memcntl

The **memcntl** system call has been added to support locking user processes' pages into memory. See **memcntl(2)** man page for more information.

setitimer

A bug has been fixed that was related to the use of the ITIMER_VIRTUAL flag/parameter with the **setitimer** system call.

4.5 Libraries

libc

- The following functions have been added to **libc** in 4.30. None of these are available in the m88kbcs Software Development Environment. See the man page entries for the descriptions.

  adjtime (system call gate)
  memcntl (system call gate)
  mlock
  mlockall
  munlock
  munlockall

- The performance of the **strcmp, memcpy, bcopy,** and **memmove** functions has been further improved.

- The **ldexp** and **frexp** functions have been rewritten to take advantage of IEEE floating point format. There is a significant improvement in performance.
- The profiling version of **libc** now provides call counts for most system calls. It does not provide call counts for the small number of assembler modules it contains. Among these are several of the **memory** and **string** functions, **bcopy, abs** and a few system calls. The **prof** command will show these functions as having time attributed to them but no call counts.
- The **sbrk** function now sets errno to ENOMEM for cases where the request would underflow or overflow in the calculation of the new break value (e.g. sbrk(-10000) when sbrk(0) == 5000).
- A bug where calling **perror** disabled future calls to **extended_perror**, has been fixed.
- The **strcmp** and **strncmp** functions now correctly handle 8-bit characters.
- The **strtol** and **strtoul** functions now correctly set the pointer position to the end of the scanned digits when the result is out of range.
- The **tzset** function will now handle POSIX timezone specifications in the TZ environment variable. POSIX timezone specifications are an extension of the traditional AT&T format. In addition, the POSIX ':' implementation defined behavior has been defined. See the man pages for **ctime(3c), zic(1M)**, and **zdump(1M)** for details.
- A bug in **printf** where %06.0f and %06.f behaved differently has been repaired. Both formats now behave identically, according to the rules in the X3.159-1989 ANSI C Standard. In particular, the 0 flag is ignored because a precision of 0 is specified in the first case, and implied in the second.
- The precision field is now ignored by **printf** when printing a NAN (not a number) value. A NAN is always printed as NaN0xdddddddd.
- Conversions done by **atof** and **printf** now use the same number of significant digits.
- In order to conform with POSIX.1, **fseek** has been modified to always **lseek**, even if the new position would have been within the buffer. This insures that the underlying file pointer position is changed.
- The **dial** function now expects to find UUCP configuration files in **/etc/uucp** rather than **/usr/lib/uucp**.
- The **initstate** function now explicitly returns a NULL pointer if the specified number of bytes of state information is less than 8.
- The bug in the **vtimes** function which caused its result to be in seconds rather than clock ticks has been repaired. The result is now scaled by HZ from **<sys/param.h>**.
- The **perror** text associated with EAGAIN has been changed from "No more processes" to the more accurate "Resource temporarily unavailable".

libc (BCS)

The BCS function entry **sys_local** is now available.

libm

- The **pow** function in **libm** has been replaced by a more accurate implementation.
- The **sqrt** function in **libm** has been replaced by a more accurate version that correctly rounds the result.

libcurses

- The performance of scrolling under **curses** has been improved. For "dumb" terminals without insert/delete line capabilities, **curses** no longer completely redraws the screen for each scroll, which typically reduces the number of output characters by 95%. "Smart" terminals show smaller benefits, typically 20% - 30%.

- The **curses** macro **unctrl** has been extended to cover any argument less than or equal to KEY_MAX.
- **Curses** functions **del_curterm** and **delterm** are now properly declared in the **/usr/include/term.h** header file.
- **Curses** function **napms** now works; previously it returned without performing the requested time delay.
- **Curses** functions **nl** and **nonl** now work in "cooked" mode.
- A performance problem has been fixed in **curses** function **setupterm.**
- **Curses** will now correctly set and reset the "local mode word" of the BSD TTY driver (documented in **tty(7)**).
- A performance problem has been fixed in **curses** function **tparm** that unnecessarily slowed the instantiation and transmission of **terminfo** sequences containing the "%c" escape on seven-bit terminal lines.
- The **curses** call **typeahead(-1)** will now always work, guaranteeing that screen updates always complete even if input is pending. Previously this call had no effect when a screen refresh was attempted from within a signal handler that interrupted a **getch** call.
- Special "keypad" characters (e.g., function key sequences) that contain characters with the high order bit set are now interpreted correctly by **curses** input routines like **getch.**
- The **addstr** family of routines in **curses** will now work for strings containing characters with the high order bit set. Previously these routines could dump core when attempting to print strings containing 8-bit characters.
- The **addch** family of routines in **curses** will now work for any **chtype** argument. Previously these routines could dump core when attempting to print **chtype** values that are overly large.
- Timeout (**halfdelay**) and unpended (**nodelay**) input methods in **curses** have been fixed. Previously **curses** input functions would hang for these modes if the screen was not completely up to date when the function was called.
- Attribute settings have been fixed in **curses.** Previously attributes would become confused for some terminals when **curses** cleared the screen.

4.6 Commands

awk

In 4.20, there was **awk** and **nawk.** In 4.30, **awk** has been renamed to **oawk,** and **nawk** has been renamed to **awk.** **Nawk** is now a symbolic link to **awk** to maintain backward compatibility with user scripts that may been have written to use **nawk.** These scripts should be changed to use **awk** because DG will drop the **nawk** symlink to **awk** as well as support for **oawk** in a future release.

cpio

- The **cpio** command now preserves major and minor device numbers.
- The **cpio** command no longer dumps core when creating multiple-tape archives with very large blocking factors.

cprs

The COFF compression utility, **cprs,** is available in this release. It reduces the size of executables built with debugging by merging multiple copies of enum, struct, and union symbols. Reductions of 30-40% are not uncommon.

csh

- The **csh** now supports filename completion (when editread is not in use). See the **csh(1)** man page for a description of filename completion.
- The **csh** now allows you to set hard limits for the **limit** built-in command.

- The **csh** now has full 8-bit character support. This feature will allow **csh** to be used with international keyboards.

df and du

The **df** and **du** commands have been modified to produce more accurate information. The **–m** option has been added to **du** to restrict its analysis to a single file system.

expreserve

The **expreserve** command now works correctly on diskless systems; the preservation directory is now located on a writable file system.

infocmp

The **infocmp** command with the **–C** option will now always generate a full **termcap** reset (**rs**) string. Previously this string could be incomplete if any of the **terminfo** reset strings (**rs1, rs2, rs3**) were unspecified.

ld

Ld now supports merging of .initp sections with output .text sections. Applications may contribute pointers to initiation routines to this section and the DG/UX 4.30 **crt0.o** will invoke each of them at program startup.

lint

DG/UX 4.30 now provides lint library source alongside the binary form (in **/usr/lib**) understood by the **lint** command. A significant number of functions have been added to the libc lint library. It now closely matches the set of functions provided by **libc**.

lpr

The BSD 4.3 line printer spooler is now included. It consists of the following commands: **lpd, lpr, lpq, lprm** and **lpc**. Administration of this spooler is *not* supported through **sysadm**, and printers can *not* be shared between the BSD 4.3 line printer spooler and the AT&T line printer spooler.

pr

The **pr** command has been fixed to correct a bug in formatting the first line of output when using the **–e** option.

REELexchange

The **REELreelexchange** utility is available in this release. REELexchange is a set of commands for reading and writing IBM and ANSI tapes. The standard tape formats are defined. See the **reelexchange_intro(1)** man page for more information.

restore

- The **restore** command now allows files to be recovered into remote-mounted file systems.
- The **restore** command now allows restoring multiple-tape dumps from remote tape drives.

sar and sadc

The **sar** command has been upgraded. A consequence is that the data file format produced by pre-4.30 versions of **sadc** can no longer be read by **sar** (and pre-4.30 versions of **sar** cannot read the output of the 4.30 **sadc**). A second consequence is that some statistics are calculated in new ways, and comparisons with old statistics may not be meaningful. The **–d** option (disk usage statistics) is now supported. See **sar(1m)** man page for more information.

script

The **script** command is now available. It is an "terminal auditing" facility that records everything that appears on the screen. See **script(1)** for more information.

dbx

The **dbx** debugger is enhanced to support the new commands **jump, position, finish, describe, save,** and **commands**, as well as extended functionality of existing commands. See the man page **dbx**(1).

spell

The **spell** command now works correctly on diskless systems; the spelling history and word list files are now located on a writable file system.

spline

The **spline** command is now available. It is used to interpolate a smooth curve from pairs of numbers from standard input. See **spline(1G)** for details.

syacload and tcload

The **/usr/lib/syac/syacload** command has been replaced by **/usr/sbin/tcload**, and the old **syac** loading procedure and support files are now obsolete.

sysadm

The **sysadm fileage** command now checks the last accessed time instead of the last modified time.

tar

- **Tar** has been upgraded to BSD 4.3.
- The **tar** command now supports the **–o** option on input. This option causes extracted files and directories to take on the user and group identifiers of the user running the program rather than those on tape.
- The **tar** command now supports archiving and retrieving files relative to the root (/) directory.

uucp

The configuration files for **uucp** have been moved from **/usr/lib/uucp** to **/etc/uucp**. When overloading 4.30, the set-up procedures will move files from the old place to the new place. If you are not overloading, be sure to restore old **uucp** configuration files to the new directory.

zdump

The **zdump** command, which deals with the new time zone compiler, is now available. See the man page for details.

zic

The **zic** command, which deals with the new time zone compiler, is now available. See the man page for details.

4.7 Facilities

editread

- The **editread** facility now synchronizes its key bindings with the line discipline special character settings. For example, reconfiguring the **intr** function within **editread** will now cause the **stty** setting for **intr** to reflect that change. The reverse is also true: using **stty** to change a line discipline special character setting will reconfigure the corresponding **editread** function (if any) to match.
- The **editread** facility will now work (in a degraded mode) even if the TERM environment variable is not set, is set incorrectly, or is set to the **dumb** terminal description.
- The **editread** facility now performs better during initialization. It makes fewer system calls and does better error checking.
- A key synonym **BS** for **<Ctrl-H>** has been added to **editread.**
- The **editread** facility now respects special line discipline modes: no-echoing mode (~ECHO) and binary input modes (CBREAK, RAW). However, while any of these modes are active, **editread** facilities will not be available.
- The **editread** facility will no longer corrupt settings for the BSD line discipline "literal next" (**lnext**) and "delayed suspend" (**dsusp**) special characters.
- The **editread** facility will now always update the display correctly for erasure operations. Previously the display line would not always be cleared properly if the terminal did not support a "clear to end of line" capability.
- The **quit** function of **editread** now works.
- The **editread** facility now restores the terminal special ("keypad") keys to "normal" mode when not at an input prompt. This makes the numeric keypad work as expected on VT100 compatible terminals.
- The key notation ^**?** has been fixed in **editread.** It once again represents the ASCII **DEL** character (0177).

terminfo and termcap

- New **terminfo** and **termcap** entries have been added to support the following terminals and modes:
  - — **d216+, d412+**, and **d462+** terminals in DG-UNIX mode
  - — VT220 compatible terminals in 132 column mode
  - — D412 and D462 terminals in 132 column DEC VT220/VT320 emulation modes
  - — **xterm** on DG AViiON work station keyboards
  - — Sun *CommandTool* "terminals"

  Consult **term(5)** for more details.
- The **termcap** database has been reorganized. It is now smaller, easier to read, and more in agreement with the **terminfo** database source for DG terminals.

- The **Terminfo xterm-fk** and **xterms-fk** entries have been fixed to properly represent the **<Prior>** and **<Next>** keys.
- The **terminfo** and **termcap d216** entries have been fixed to present the correct number of function keys.

5. Notes and Warnings

This section describes notes and warnings for Release 4.30 of the DG/UX System.

5.1 General

Graphics terminals

- For the **avx300** and **avx320** graphics terminals you should set the TERM variable to **avx300** in your local **.profile** and/or in the **/etc/profile** file.
- Also, to use the "backspace" key as the delete key set the stty erase parameter in the **/.profile** and **/admin.profile** files to ^h instead of ^?.

Data loss on a parallel printer

If you are experiencing the following problems on a printer attached to the main parallel port:

- Loss of characters on printed output.
- File or document leaves print queue but never prints.

you should obtain ECO #DC35 from your DG representative. Your system board will receive the ECO and your (DG only) printer will receive a termination resistance adjustment.

If your printer is not a DG product, call your service representative for that product and have him adjust your printers termination resistance values as follows: 470 ohms to +5V; 470 ohms to GROUND. This will most likely involve diagonal clippers and a soldering iron and is not recommended as a "do-it-yourself" modification as it may void your warranty.

Bootstrap

- The bootstrap implementation was changed to work around a SCSI controller chip bug. Due to this change, you may notice the bootstrap to be slower on AViiON work stations than it was in previous releases.
- Infrequently, when booting a system, the bootstrap will panic or hang. This occurs because the bootstrap program pulls an uninitialized data structure from memory. If this occurs, simply re-boot.

Overloading

- The prototype **crontab** file for the root user has changed in this release. If you are overloading a previous release and already have a **crontab** file for root, the new prototype file **/var/spool/cron/crontabs/root.proto**, will not get installed automatically. You will want to merge the new features of the prototype file into your root **crontab** file, **/var/spool/cron/crontabs/root**. The new features add some safety features to the removal of old temporary files.
- A small number of other prototype files have been changed in this release. During setup, the base file is saved by renaming it with an appendix on the base filename and then the new prototype is installed as if the base file did not exist. A message will be displayed on the console during setup whenever this occurs. If there are site-specific changes in the old base file, the system administrator must merge the changes into the new base file. The system configuration file prototype is one prototype that has changed. Do not attempt to build a kernel using an old system file. To help prevent you from doing this, the setup procedures rename all existing system files in **/var/Build**.
- The *Installing and Managing the DG/UX System* manual should be consulted about logical disk size requirements before attempting to overload with 4.30. In particular, the **usr** file system size is reduced from 200,000 blocks to 160,000 blocks in 4.30.

- All products should be upgraded to the latest release at one time to insure proper operation of your system. For example, TCP/IP Release 4.20 should be upgraded to its 4.30 release at the same time that DG/UX is upgraded to Release 4.30. The products that have this dependency are TCP/IP, NFS, Gnu C, X Windows, and OSF Motif.

UUCP Directories

If you use **uucp**, please set the modes of directories in **/var/spool/uucp** (inclusive) to 777 using the following command:

```
find /var/spool/uucp -type d -exec chmod 777 {} \;
```

5.2 Internationalization

Keyboards

In order to use an international keyboard on the console, the proper character set must be selected from the SCM "Change console parameters" menu before booting DG/UX.

Commands

- The **cpio, date, ls, pr**, and **sort** commands provide the date and time in the language and format determined by the value of the **LANGUAGE** environment variable. While the United States conventions remain the default, other languages can be supported by creating and installing a file for the language desired in the **/lib/cftime** directory. The content of that file includes: month and weekday names (full and abbreviated), default local time, date, pre-noon and post-noon formats, and the default output of the **date** command if the CFTIME environment variable is not set. In addition, time zones and alternate time zones (such as daylight time) can be defined in terms of hours and minutes using the TZ environment variable. See **date(1)** and **environ(5)** for more information.
- The **wc** command does not include a word made up entirely of 8-bit characters in its word count.
- The **file** command reports that files containing 8-bit characters are data files rather than ASCII files.
- In order to use 8-bit characters in the subject of a **mailx** message, the subject should be left blank, and a line containing **~s<subject>** should be included in the body of the message.
- The **sysadm** utility does not support usernames, hostnames, printer names, or file names containing 8-bit characters.
- The **lpadmin** command does not accept arguments containing 8-bit characters.
- When in insertion mode, the **vi** command will display the octal code for any characters which are not in the current character class (default is United States).
- The **curses** library uses **ctype** macros to classify characters as printable or not during output operations. Programs should therefore call the **setchrclass** routine (before calling **initscr**) to ensure correct display of all text.

SCM

The SCM emulator does not display or allow users to generate 8-bit characters.

Terminal emulator and xterm

The generation of characters which do not have corresponding keys on the current keyboard is not supported by either the kernel terminal emulator or by xterm.

Terminals

- The D216 and D216+ terminals in VT100 mode support only 7-bit characters, since 8-bit support is not available on actual VT100 terminals.
- The D412 and D462 terminals in VT220 mode, and the D412+ and D462+ terminals in VT320 mode, use the DEC multinational character set, which differs slightly from the ISO 8859-1 character set used by DG/UX. This means that in order to generate several of the 8-bit characters, a compose key sequence as described in Appendix C of the D216/D216E and D412/D462 Display Terminals User's Manual is required.

istrip

The default gettydefs setting for the ISTRIP characteristic will cause all characters to be stripped to 7 bits on input. Users must either change the gettydefs entries to -ISTRIP or issue **stty –istrip** upon login to the Bourne shell, or **stty pass8** upon login to the 'C' shell in order to display 8-bit characters.

5.3 Kernel

/etc/inittab entries

When configuring your kernel (**sysadm newdgux**), please make sure that the value for **NPROC** is at least as large as four times that of the number of lines (entries) in **/etc/inittab**. This will provide adequate process table space for systems with a large number of terminals.

tty renumbering

The association between physical and logical tty connections now depends on the order of appearance of **duart** and **syac** in the kernel configuration file. In previous releases, the **duart** connections were always tty0 (and tty1 if you had two duarts configured) even if entries for **syac** appeared first in the configuration file.

Memory-based file systems

Support has been added for memory-based file systems. See **mfs(4)**. Also observe the following warnings when using memory file systems:

- Be careful not to overcommit the swap space available to the system. Because of the way DG/UX allocates memory, if you create a large memory file system, run some large application, then fill the memory file system, you may exhaust the swap space on the system. This will cause the system to kill arbitrary processes in order to recover the swap space.
- Do not use the use_wired_memory option on diskless work stations; it can cause system failures.

Swap space

Beginning with DG/UX Release 4.30, user processes are charged for swap space as they allocate virtual memory (e.g., with **malloc, brk**, or **fork**) rather than when the swap space is actually used. This prevents overcommitment of the available swap space. However, programs which ran successfully under previous DG/UX revisions may now fail attempting to allocate virtual memory. If this occurs, "out of paging area space" diagnostics will appear on the system console. In this event, correct any programs which are known to allocate virtual memory that is never used. Alternatively, allocate more swap space for your system; we recommend a minimum of 50,000 blocks.

Emergency system shutdown

An emergency system shutdown can be performed from the system console, which is either a terminal on the duart controller or the work station graphics console. The syscon device driver will panic the system with panic code

```
030000013 (TS_PANIC_OPERATOR_SHUTDOWN)
```

when it detects the following hot key character sequence: ^]^[^]^[^]^[, where the caret (^) indicates the next character (']' or '[') is typed while holding down the <CONTROL> key. These must be typed in order with no intervening characters.

For "soft" system hangs, where the console driver is operative, the hot-key sequence can be used to initiate a system dump. To initiate a system dump from a "hard" system hang, reset the system using the reset switch then enter:

```
S 1000
```

at the SCM prompt. Note that this cannot be used to produce a system dump from a system that hangs after the panic sequence has been entered.

With a diskless work station: In case **inen()** did not work as a dump destination device, a dump may still be performed by rebooting the machine with a -d flag at the SCM prompt with the following command:

```
b inen()dgux -d
```

SCSI devices

- See **st(7)** and **sd(7)** for a list of device model numbers to which the following information pertains.
- SCSI devices can be auto-configured by specifying "*" in the SCSI ID field of the DG/UX system file device specification. If SCSI ID "*" is used, e.g. **sd(insc(),*)**, the system will configure all instances of the specified class of device present on the SCSI bus.
- The system may hang while configuring 662-megabyte SCSI hard disks. This is a problem with the firmware on the MAXTOR hard disk and is fixed in rev DG04 firmware. Contact your Data General representative for upgrading your MAXTOR disk's firmware.
- When a SCSI tape encounters an error writing a filemark while closing the device, the tape will not be rewound to Beginning of Tape (BOT) even if it was opened with the "rewind-on-close" option.
- While using the 2 GB mass storage device on an AViiON Series 300, 400, or 4000 system, the DG/UX system may very infrequently panic with code 053000023.
- The 2 GB mass storage device is intended only for use as an archiving system under streaming conditions. It is **not** supported as a means to dump physical memory after a system panic or hang.
- ENXIO errors may occur infrequently in user programs which use ioctl's to space backwards over filemarks. This problem is due to a bug in the 150MB cartridge tape firmware and therefore this condition can also exist in systems which shipped with prior releases of DG/UX. If this error occurs, you may want to contact your Data General representative for upgrading your tape drive firmware.

Synchronous SCSI protocol

Previous releases of DG/UX do not support SCSI synchronous transfers on the AV5000/AV6000 Series of servers due to a controller firmware limitation. Synchronous SCSI transfers provide a performance improvement over asynchronous transfers. Revision 11 firmware for the VME SCSI controller removes this limitation. During system boot, the controller revision is checked and if it is not Revision 11, the message:

```
Firmware in SCSI controller "cisc()" is out of date --

see release notice.
```

is printed on the system console. This message indicates that the current controller firmware will not support synchronous transfers. There is no risk in using firmware revisions prior to revision 11 since asynchronous SCSI transfers will continue to work. However, upgrading to Revision 11 firmware may provide a performance benefit if any of the following SCSI disks are in your system: 179-megabyte SCSI disk, 662-megabyte SCSI disk or 322-

megabyte half-height SCSI disk. Contact your Data General representative for upgrading your SCSI controller firmware.

Work station graphics console terminal emulator

Delete, tab, and caps lock support is provided in the work station graphics console terminal emulator, though the Caps Lock LED does not work correctly. No other special keys are supported.

Diskette devices

Hardware notes:

- The (pre-4.30) restriction of mixing different floppy types in a single chassis has been removed with this release.
- The 3.5" diskette device supports 720-kilobytes and 1.44-megabyte formats. The 5.25" diskette device supports 360-kilobytes, 720-kilobytes, and 1.2-megabyte formats. Obtain pre-formatted diskettes from your DG representative. Alternatively, if you have DG RBOS (revision 4 or later) capability, you can format your own floppy diskettes. RBOS is **not** a DG/UX utility.
- The 3.5" diskette device may be either unit 0 or unit 1. The 5.25" diskette device may be unit 0, 1, 2, or 3. The 5.25" must be set to unit 2 or 3 in a mixed configuration.
- To use a diskette device, you must set the SCSI ID, the unit number, and the adapter card straps. The adapter card has the SCSI ID settings and the strap settings, and the device itself has the unit setting.
- To use a 3.5" diskette device, set the strap settings on the TEAC adapter card to H, F, LEV, STL, and PAR. To use a 5.25" diskette device, set the strap settings on the TEAC adapter card to G, F, LEV, STL, and PAR. The H is for 1.44-megabyte format, G is for 1.2-megabyte format, and F is for 720-kilobytes format. In a mixed configuration, the straps should be set as described for the 3.5" diskette device.

Software notes:

- **Sysadm** dumps to tapes only and does not recognize diskette devices.
- Before removing a diskette serving as a mounted file system, you should unmount the file system and deregister the device; otherwise, a file system error about a "sealed LDU" or the message "LDU is no longer fault tolerant" will appear. If this happens, you need to run **fsck** to use the diskette device again.
- When accessing a diskette device with **tar, cpio, dump, dump2**, or **dd**, refer to the device by its node in the directories **/dev/rpdsk** or **/dev/pdsk**.
- Because diskettes do not have tape marks (as found on magnetic tape), use the **skip** option when using **dd** to access a diskette device.
- You can exchange **cpio**, **tar**, and **dd** files between 386/ix 2.0 and DG/UX 4.20 and later systems using 1.2-megabyte and 360-kilobytes formats for 5.25" diskettes and 1.44-megabyte and 720-kilobytes formats for 3.5" diskettes. Hardware limitations prevent 386/ix systems from reading 720-kilobytes format 5.25" diskettes.
- When you use **cpio** to create files that you intend to move from a DG/UX system to a 386/ix system, use the **–c** option. For files that you intend to move from a 386/ix system to a DG/UX system, the **–c** option is not necessary. The default block sizes for **tar** on 386/ix and DG/UX 4.30 systems are compatible.
- If an unformatted diskette is in the diskette drive during configuration, the system will print a "Physical disk information table error 4026027" message. This message simply means that the file system attempted to read information from the diskette, but failed due to the fact that it wasn't formatted.

Diskman

- When booting stand-alone **diskman** and entering devices for a graphics work station, enter only the **grfx** and the **kbd** devices, and for a server system enter only the **duart** device.
- **Diskman** uses control-H, which is <backspace> on the work station keyboard, instead of <delete> for deleting characters.

Starter kernel

- For the work station, the **grfx** and **kbd** devices must be configured when booting the starter kernel, **/dgux.starter**, if using the graphics console. For the server, or a work station using an async monitor console, the **duart** device must be configured. These devices are in addition to the tape, disk, and other devices you may need.

Pre-configured kernels

When executing programs that use the M88100 'XMEM' instruction, there is a chance that the process may loop forever executing the XMEM instruction; the process will not be killable when this occurs. The bug is fixed in the dgux kernel libraries included in the 4.30 release. Should you encounter this problem running any of the **preconfigured** dgux images shipped with the 4.30 release, simply build a new dgux executable image (**sysadm newdgux**) and reboot the system.

Prom requirements

Kernel and **diskman** images with this release require work station PROM revision 3.00, or greater, and a server PROM revision greater than 0.00. Servers with expansion memory require PROM revision 3.00 or greater.

system.aviion

The template system file created by **sysadm newdgux** (**/usr/src/uts/aviion/Build/system.aviion**) is only an example of devices and others, such as **sd(cisc(),*)**, will have to be added to allow for all SCSI disk devices.

PROM bug

A bug has been identified in the work station PROM (all revs). When booting from the **inen** network device with a pathname specified and that path contains a period, the first stage (PROM) boot fails. The PROM incorrectly assumes that a field containing a period indicates that the field is the internet address of the server to use.

The problem can be avoided by supplying the internet address field (don't forget the colon delimiter) when booting with a pathname that contains a period. Alternatively, rename the boot path to eliminate the period.

Tape driver

The behavior of the raw mag-tape interface (**/dev/rmt**) has changed. In previous releases, if a tape file was opened with write intent, a file mark was written upon closing the file only if data was actually written to the file. This is no longer true. Upon closing a file opened with write intent, a file mark is *always* written. This change was made to conform with the 88open BCS specifications.

Panics

A kernel panic with panic code 01000026 may occur if the work station network driver, **inen**, is configured into a **server** kernel.

/usr/release/dgux_4.30.panic.codes has a list of panic codes and sub-system code. This may be useful in determining the cause of a panic.

5.4 Header files

assert.h

The #ifdef surrounding the definition of **assert** should have used the __STDC__ macro rather than the _USING... macro. This will result in compile errors when compiling in traditional mode and defining the _POSIX_SOURCE macro. This will be fixed in a future release. If you have problems with this, contact your DG representative for a patch.

malloc.h and stdlib.h

If malloc.h and stdlib.h are included in the same source file, and you are compiling in ANSI mode, there will be conflicting definitions for: **malloc(), free()** and **realloc()**. malloc.h contains the traditional ATT declarations of the above library routines, and stdlib.h contains the ANSI declarations.

If you are using ANSI mode, then you should be using the **malloc()** declarations in stdlib.h and not include malloc.h. This conflict will be repaired in a future release.

5.5 System Calls

wait

Using the **wait** system call with no arguments is a degenerate usage, and it has never been supported, but since it usually functioned as intended, the usage has become idiomatic. However, the usage cannot be relied upon. Replace all occurrences of **wait()** with **wait(NULL)**.

5.6 Libraries

libc

The profiling version of **libc** does not provide call counts for the small number of assembler modules it contains. Among these are several of the **memory** and **string** functions, **bcopy, abs** and a few system calls. The **prof** command will show these functions as having time attributed to them but no call counts.

malloc and sbrk issues

- Applications which make use of any C library functions should not call **sbrk** directly. Any C library function may call **malloc**, which relies on the break value remaining the same between invocations. Applications that use C library functions should use **malloc** for memory management.
- Sometimes application developers like to substitute custom routines in place of standard library routines. You should be aware this behavior is generally discouraged by ANSI C, and we recommend against it. However, some existing applications already provide custom code for **malloc**, for example.

  In DG/UX 4.20 and 4.30, **libc** and **libmalloc** are constructed such that each function, **<name>**, is implemented as an object module containing a single jump instruction to the function **<_name>**. The function **<_name>** is in a separate object module which contains the code that implements the function.

  This means that if you are supplying a routine that you want to have a global effect at link time, e.g. a **malloc** routine, you need to supply the **_malloc** function in an object module on your **ld** command line. Your code will reference **malloc**, which pulls in the module in **libc** that creates the linkage from the name **malloc** to your object module that provides the function **_malloc**. In this example, if you do not supply the **_malloc** routine, then all library routines that use **malloc** would use the standard **malloc** and your program would have a private **malloc** which could cause unpredictable results.

  If you are supplying a routine that you want to have a private effect at link time, you need not supply the **_name** function as in the example above. However, having an application use a different **malloc()** from the one used by

libc functions is an invitation to inconsistent memory allocation and program failure, and perhaps system panic.

cuserid

Beware of the current **cuserid** functionality. This is *NOT* a SVID compliant function! This function is implemented with POSIX semantics and returns the name associated with the *effective* user ID of the process, not the *real* user ID. Programs which change their effective user id (e.g. setuid programs) and use **cuserid** could create a serious security hole if compiled with this implementation. To use the SVID compliant **cuserid**, use the function name **sysv3_cuserid**.

5.7 Commands

as and ld

- The assembler and linker now produce object files in OCS-compliant format. The symbols with storage class C_VERSION are no longer produced, although they are handled correctly for compatibility with DG/UX 4.20. Object files now have 32 bit line numbers as specified by the OCS.
- The linker supports a new option **–n** which causes no contributions to sections in the output that were not in the input. In particular it makes no contributions to .init, .tdesc, or .initp. Use of this option with almost any object built by a compiler will produce a faulty executable without use of a linker directives file.
- By default, **ld** now merges .init, .tdesc, .initp, and .finip sections into the output text section. Link edit scripts, if they are used, must explicitly merge these sections into the text section for correct operation of debuggers and other DG/UX utilities.
- Some nonstandard programs dereference NULL pointers and expect address 0 to contain 0. These programs can be made to work by using the following link editor script. The script name may be placed anywhere on the **ld** command line.

```
SECTIONS
{
    .text  BIND(0x0) BLOCK (0x10000): { . += 0x10;
                                        *(.init)
                                        *(.text)
                                        *(.tdesc)
                                        *(.initp)
                                        *(.finip)
                                      }

    GROUP BIND( NEXT(0x10000) +
                ((SIZEOF(.text) + ADDR(.text) ) % 0x10000)) :
        {       .data   : { }
                .bss    : { }
        }
}
```

cu and uucp

- **Cu** and **uucp** have supported an undocumented option in the **Devices** file. It is now documented in the 4.30 manuals. In short, a **,M** option immediately after the line specification (i.e., *tty01,M*) will tell **cu** and **uucp** oriented commands to open the device with O_NDELAY and then clear it after opening it. This is useful for initiating a connection to modems where a generic open timeout is encountered without the **,M** option.
- Many of the configuration files that once resided in **/usr/lib/uucp** have now been moved to **/etc/uucp**. This was done because of the need for diskless systems to have their own private copies of the files.

If you use **uucp** and you are overloading a previous DG/UX release, the files in **/usr/lib/uucp** will be renamed to "*_4.20". You will have to sort through these renamed files and merge them into the correct file names in **/etc/uucp**. During setup (sysadm setuppackage) of DG/UX you will see messages like:

```
Saving $USR/lib/uucp/Dialcodes as Dialcodes_4.20
```

There will be a "README" file in **/usr/lib/uucp** to remind you that you will need to merge into /etc/uucp.

The uucp files that have changed location are:

Cvt
Devices
Dialcodes
Dialers
Maxuuscheds
Maxuuxqts
Permissions
Poll
SetUp
Sysfiles
Systems
remote.unknown
uudemon.admin
uudemon.cleanup
uudemon.hour
uudemon.poll.

mxdb

Under certain circumstances, processes run under the mxdb debugger will be unexpectedly killed. This is most likely to happen when single-stepping across load/store instructions on very heavily loaded systems.

dbx

When using **dbx** with FORTRAN executables (ghf77), you may experience difficulty in examining common block data. The problem manifests itself with the error:

```
Can't find symbol x
```

The fix is to obtain revision 1.8.5 of the Green Hills FORTRAN compiler. With it you will also receive a dbx that will support it.

crash

The **crash** command has been re-implemented for DG/UX release 4.30. See the **crash(1m)** man page for details.

lpsched

The lp scheduler can unlock its lock on the outputq file before it is done manipulating the file. If the lp program is waiting for the lock to be released, it starts using the file before the scheduler is done with it. This contention causes the lp system to fail, and requires a restart of the scheduler to get it repaired. If you experience this problem, contact your DG representative for a patch.

tar

tar with the -u option will hang the system.

cron

cron does not allow all forms of output redirection when the command includes invocation of **csh**. A **cron** command in the form of:

```
0 * * * * /bin/csh -c "echo test of redirection > /tmp/test"
```

will fail. Shell scripts will exit with status '1' and executables (as /bin/csh in above example) will dump core, as soon as they try to write to the output file. If the redirection occurs **outside** of the quotes as in:

```
0 * * * * /bin/csh -c "echo test of redirection" > /tmp/test
```

everything will work correctly. This is an opposite behavior from 4.20 and should be noted since you may have crontabs that have this restriction.

The supplied file **/var/spool/cron/crontabs/root.proto** contains the following line:

```
0 4 * * * /bin/su - adm -c "/usr/lib/acct/runacct 2> /usr/adm/acct/nite/fd2log"
```

If user **adm** is changed from using **sh** to **csh** as its default shell (by editing /etc/passwd), then change the above line in **root**'s crontab to:

```
0 4 * * * /bin/su - adm -c "/usr/lib/acct/runacct" 2> /usr/adm/acct/nite/fd2log
```

5.8 X Windows

xstart and xdm

The X window system should be started using the **xstart** or **xdm** commands. Starting X via other methods may cause incorrect behavior.

6. Documentation

6.1 Titles

The following documents are available for release 4.30 of the DG/UX system:

Publication	Part Number
AViiON DG/UX System Release Notice	085-600125-03
GNU-C Release Notice	085-600128-02
DTK Release Notice	085-600139-02
Learning the UNIX Operating System	069-701042-00
Using the DG/UX System	069-701035-01
Using the DG/UX Editors	069-701036-01
Installing DG/UX on an AViiON Workstation with a Hard Disk	069-000520-00
Installing and Managing the DG/UX System	093-701052-02
Using DG/UX Software Development Tools	093-701078-00
Programming in the DG/UX System Applic. Environment	093-701076-00
Using the DG/UX System Kernel Debugger	093-701075-00
Porting Applications to the DG/UX System	069-701059-02
Writing a Device Driver for the DG/UX System	093-701053-03
User's Reference for the DG/UX System	093-701054-01
System Manager's Reference for the DG/UX System	093-701050-01
Programmer's Reference for the DG/UX System (V. 1)	093-701055-01
Programmer's Reference for the DG/UX System (V. 2)	093-701056-01
STREAMS Primer for the DG/UX System	069-701033-00
STREAMS Programmer's Guide	069-701034-00
Using DTK on the DG/UX System	069-701039-00
DTK Technical Summary for the DG/UX System	069-701041-00
Binary Compatibility Standard	069-701043-01
Portable Operating System Interface (POSIX)	069-701045-00
POSIX.1 Conformance Document for the DG/UX System	069-701078-00
Object Compatibility Standard	069-701044-01
C: A Reference Manual (Harbison/Steele)	069-100226-00
Using GNU CC (Stallworth)	069-100317-01
Overview of the DG/UX Applic. Environment*	069-701082-00
Programming in the Kernel Environment*	093-701083-00

* These manuals will be available in the Fall of 1990. In addition, "Writing a Device Driver for the DG/UX System" will undergo a major restructuring and be available in the Fall of 1990 under the new part number 093-701085-00.

6.2 Changes

There are extensive changes to chapter 2 of *Installing and Managing the DG/UX System.* Make sure you have this **before** installing release 4.30.

7. Software Distribution

7.1 Media

DG/UX is bundled within the **Operating System User's Package.**

1. Model Numbers: Q001A P001A

7.2 Organization

The **Operating Systems User's Package** tape has the following layout:

File	Name	Size	Type	LoadPoint	Options
0	boot.aviion	226	image		
1	xdrtoc	6144	toc		
2	standalone	5252	image		
3	dtoc	512	image		
4	dgux__r.dirs	5252	tar	/	required movable
5	dgux__urp.dirs_5	49152	tar	/tmp/root.proto.tmp	required movable
6	dgux__r.bin	16384	tar	/	required movable
7	dgux__urp.bin_7	16384	tar	/tmp/root.proto.tmp	required movable
8	dgux__r.ucmds	1049600	tar	/	required movable
9	dgux__urp.ucmds_9	1049600	tar	/tmp/root.proto.tmp	required movable
10	dgux__r.sysadm	16384	tar	/	required movable
11	dgux__urp.sysadm_11	16384	tar	/tmp/root.proto.tmp	required movable
12	dgux__r.krn	1328128	tar	/	required movable
13	dgux__urp.krn_13	197632	tar	/tmp/root.proto.tmp	required movable
14	dgux__r.ucmds+	197632	tar	/	required movable
15	dgux__urp.ucmds+_15	197632	tar	/tmp/root.proto.tmp	required movable
16	dgux__r.krn+	1147904	tar	/	required movable
17	dgux__urp.krn+_17	16384	tar	/tmp/root.proto.tmp	required movable
18	dgux__u.dirs	132096	tar	/usr	required movable
19	dgux__u.sysh	1065984	tar	/usr	required movable
20	dgux__u.krn	8032256	tar	/usr	required movable
21	dgux__u.debug	640000	tar	/usr	required movable
22	dgux__u.ulibs	3377152	tar	/usr	required movable
23	dgux__u.plibs	3196928	tar	/usr	required movable
24	dgux__u.ucmds	23179264	tar	/usr	required movable
25	dgux__u.sysadm	1606656	tar	/usr	required movable
26	dgux__u.sdb	312320	tar	/usr	required movable
27	dgux__u.tcpip	361472	tar	/usr	required movable
28	dgux__u.fbe	1786880	tar	/usr	required movable
29	dgux__u.dbx	640000	tar	/usr	required movable
30	dgux__u.codes	115712	tar	/usr	required movable
31	dgux__u.release	181248	tar	/usr	required movable
32	dgux__u.krn+	2721792	tar	/usr	required movable
33	dgux__u.krn++	4803584	tar	/usr	required movable
34	dgux__u.krn+++	2689024	tar	/usr	required movable
35	dgux__u.ulibs+	1983488	tar	/usr	required movable
36	dgux__u.plibs+	2246656	tar	/usr	required movable
37	dgux__u.ucmds+	3721216	tar	/usr	required movable
38	dgux__u.sysadm+	65536	tar	/usr	required movable
39	dgux__u.fbe+	16384	tar	/usr	required movable
40	dgux__u.C.sysadm	16384	tar	/usr	required movable
41	dgux__u.D.uucp	16384	tar	/usr	required movable
42	dgux__u.F.tcload	16384	tar	/usr	required movable
43	dgux__u.G.config	98304	tar	/usr	required movable
44	dgux__u.I.loadpkg	16384	tar	/usr	required movable
45	dgux__u.E.ts	16384	tar	/usr	required movable
46	dgux__u.release+	16384	tar	/usr	required movable

47	dgux__fl.name	16384	tar	/usr	required movable
48	dgux.man__u.man	3803136	tar	/usr	required movable
49	dgux.man__u.man	32768	tar	/usr	required movable
50	dgux.man__fl.name	16384	tar	/usr	required movable
51	dtk__r.img	49152	tar	/	required movable
52	dtk__u.img	3180544	tar	/usr	required movable
53	dtk__fl.name	16384	/usr	required movable	
54	dtk.man__u.man	115712	tar	/usr	required movable
55	dtk.man__fl.name	16384	/usr	required movable	
56	gcc__u.img	2443264	tar	/usr	required movable
57	gcc__fl.name	16384	/usr	required movable	
58	gcc.man__u.man	32768	tar	/usr	required movable
59	gcc.man__fl.name	16384	/usr	required movable	

7.3 Files

A list of files that are loaded when the DG/UX system is loaded on your disk is available in a file called **/usr/release/dgux_4.30.fl**. **Sysadm** is used to load the C compiler and DTK. File lists in **/usr/release** are called **gcc_1.35.20.fl** and **DTK_2.10.fl**.

8. Installation Instructions

See chapter 2 and Appendix F of *Installing and Managing a DG/UX System.* There are extensive changes to chapter 2 of *Installing and Managing the DG/UX System.* Make sure you have this **before** installing release 4.30. Appendix F contains important information for diskless servers as well as other important **overloading** issues.

Client Setup Warning

Do not attempt to set up your diskless client **root** space from the server. Although that option is made possible via **sysadm setuppackage**, you should **only** setup **usr** space for diskless clients from the server. The diskless clients will have to set up their own roots (**sysadm setuppackage**). This restriction will be lifted in a future release.

9. Preparing a Software Trouble Report (STR)

If you believe you have found an error in the DG/UX System or its documentation, or if you have a suggestion for enhancing or improving the product, use a Data General Software Trouble Report (STR) to communicate this to DG.

The standard STR form is available once DG/UX is loaded. It is called **/usr/release/STR_form** and is line printer ready. STR forms are also available from the nearest DG office or DG representative, or the Software Support Center. If your contract permits, you may report the information called for in this section to your Data General representative.

9.1 Gathering STR Information

To help us expedite STR processing, include only one problem or suggestion on each STR form. Please follow these guidelines when filling out your Software Trouble Report:

1. List the name of the product as the "DG/UX", model number Q001A P001A. List the release number as 4.30; if you are running an update or patch, then include its number as well.
2. Decide what kind of STR you are writing:
   - Enhancement: describe the proposed enhancement clearly and tell why you want it. The better we understand your desire, the easier it is for us to evaluate your request.
   - Documentation Error: list the title and part number of the document and list the page and paragraph (or section) containing the error. Please state exactly why you think there is an error.
   - Software Problem: clearly and specifically state the problem so that support personnel can try to reproduce it. See the section *Software Problems* below for more details.
3. On the STR form provide all of the following information:
   - Date
   - Name and release number of the product
   - Release of the operating system
   - CPU type
   - Hardware configuration (if relevant)
   - Names and release numbers of other software running on the system
   - The command line or scenario that caused the problem
   - The action(s) necessary to reproduce the problem
   - How often the problem occurs and how serious it is
4. If the problem occurred soon after installing a new release of software or new hardware, please note this.
5. If you received an error message, please write down the exact text (and number, if present) of the message.

9.2 Software Problems

Report any particular activity or program running on the system that seems to cause the problem. If the program is supplied by DG, report in detail the exact steps used to reproduce the problem. If the program is supplied by another vendor or written by an installation, include a copy of the program and its source code if possible. **Again, report in detail the exact steps used to reproduce the problem.**

9.2.1 System Panics

If your system panics, be sure to **record the panic number**. Then take a dump of the system memory as described below in *Taking System Dumps*.

9.2.2 System Halts or Hangs

If the system hangs:

1. Try the ''hot key'' sequence (see the *Notes and Warnings* section of the DG/UX System Release Notice).
2. Reset your machine using the reset key or switch. Once in the SCM, enter "**reset**" followed by "**start 1000**". This will invoke the dump sequence. See *Taking System Dumps*.

9.2.3 Incorrect Behavior From a System Call or Device Driver

Write the **shortest possible** program that demonstrates the problem. This can be a shortened version of your actual program, without unnecessary variables and logic. Supply both source and executable copies of this program and a copy of the system image (**/dgux**) with a detailed description of how to reproduce the problem.

9.2.4 Incorrect Behavior of a Command

Write the **shortest possible** shell script that demonstrates the problem. Supply a copy of this script with a detailed description of how to reproduce the problem. Before submitting the problem, be sure that it can be duplicated under ''sterile'' conditions; i.e., with a standard path, standard permissions on files, etc. In other words, try to ensure that your environment is not the cause of the problem.

9.3 Taking System Dumps

If your system panics, you will have the opportunity to take a dump of system memory. **In all cases**, when you provide a dump, you should also **provide a copy of the tailored system image** (usually named **/dgux**) that was running at the time of the crash. This image contains vital information necessary for interpretation of the memory dump; the memory dump is useless without the system image. See *Notes On Tapes* below for details on tape format.

Upon system panic, you will automatically enter the system dump dialog. The particular dialog depends on the type of dump device. The dump device can either be a local tape device, or, for a diskless AViiON work station, a network device. Each of these scenarios is described below. In either case, the dialog begins as follows:

```
Do you want to take a system dump? [Y] <NL>
```

Press the New Line key to answer yes, and continue with the appropriate dialog (tape or network) described below.

9.3.1 Dumping to a Local Tape Drive

You will be prompted for the dump device in DG/UX common device specification format. The example below is for the common case of a SCSI tape drive on SCSI ID 4 of a work station's integrated SCSI controller. Substitute the name of the tape device you wish to use on your system.

```
Dump destination device? st(insc(),4) <NL>
```

You will then be prompted to mount a tape:

```
Mount tape.  Type newline when tape is ready.
```

If the system memory image is too large to fit on one tape, you will be prompted to load subsequent tape volumes:

```
Tape volume 1 completed.
Mount tape.  Type newline when tape is ready.
```

If any tape volume write fails, you will be allowed to restart the dump at the beginning of that volume:

```
Hard error on tape volume 1.  Restarting volume from checkpoint.
Mount tape.  Type newline when tape is ready.
```

The final volume will be rewound upon completion of the dump:

```
System dump completed successfully.
```

9.3.2 Dumping to a Network Device on a Diskless Work station

You will be prompted for the dump device in DG/UX common device specification format. The example below is for the common case of an integrated Ethernet controller on a diskless work station. Substitute the name of the network device you wish to use on your system.

```
Dump destination device?  inen() <NL>
```

The system bootstrap will be executed. A dump will be taken before the kernel is rebooted. The dump will be written to the file described as the dump target in the work station's boot parameter entry in the server's **/etc/bootparams** file. It may also be specified in the YP version of bootparams (Try **ypcat bootparams**). An excerpt from that file for a client named *workstation* would look like this:

```
workstation root=server:/srv/release/PRIMARY/root/workstation \
            swap=server:/srv/swap/workstation \
            dump=server:/srv/dump/workstation
```

The resulting dump will not have the kernel text in it, so the image of the kernel that was running when the dump was taken must be supplied along with the dump itself. Note: the dump file must exist on the server **/srv/dump/workstation**) and the dump file must be exported with root access by the server or the dump will fail. An estimate of the amount of time required to create the dump file is printed on the console.

If the dump file cannot be written due to permission problems or lack of file system space, the dump can be restarted by rebooting the machine from the SCM> prompt using the **-d** (dump) flag:

```
SCM>  b inen() –d <NL>
```

Upon completion the system will be rebooted using the default system image.

9.4 Notes on Tapes

Tapes containing a system memory dump should be clearly labeled as such. Use the following format for cartridge tapes:

Tape 1 file 0: memory contents (final volume) in memory dump format
Tape 1 file 1: system image (normally **/dgux**) in **cpio** format
Tape 1 file 2: other files, programs, etc., in **cpio** format

Do not use absolute pathnames (i.e., starting with /). The following example shows the steps for making such a tape on a system whose primary tape drive is **/dev/rmt/0**. Use the device name appropriate for your system.

1. For file 0 on tape 1: Dump the system memory as described above. The final tape volume will be rewound upon completion of the dump.

2. For file 1 on tape 1: Use a command line like this:

   # mt –f /dev/rmt/0n fsf; cd /; echo dgux | cpio -ohBcv > /dev/rmt/0n

   The tape will not be rewound and will be positioned for any additional files to be written to tape.

3. For each necessary file after file 1 on tape 1: Use a command line like this:

   # ls *filenames* **| cpio -oBcv > /dev/rmt/0**

   The tape will rewind after this command line.

For problems that do not involve a system dump, please put all files associated with the problem on tape file 0 in **cpio** format, using this command:

 # ls *filenames* **| cpio -oBcv > /dev/rmt/0**

Please place a label on the tape clearly indicating its contents.

--- End of Release Notice ---