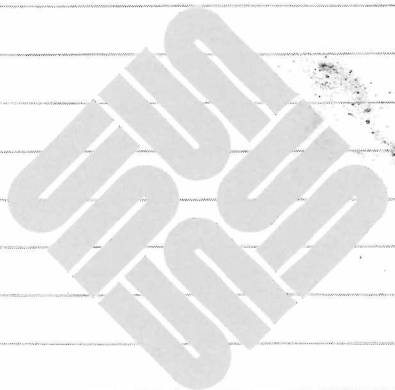
sun microsystems

System Summary and Change Notes *for the* Sun Workstation



Sun Microsystems, Inc. • 2550 Garcia Avenue • Mountain View, CA 94043 • 415-960-1300

Part No: 800-1178-01
Revision A of 15 May, 1985

Release 2.0 System Summary

Contents

Introduction

Welcome to the Sun Workstation‡ and its operating software based on the UNIX† operating system.

New Customers Please Note
that these notes provide a description of the major features of this release and changes since the previous release. If you have just received your first shipment of a Sun Workstation‡, the first manual you should read is ***Installing UNIX on the Sun Workstation***, which contains a detailed explanation of how to set up the Sun Workstation and how to install the operating software on it. The manual entitled ***System Administration for the Sun-2 Workstation*** covers details of day-to-day administration and system operation.

The Sun software is based upon the system known as 4.2 BSD — a version of the UNIX operating system for the DEC VAX machine family, with enhancements from the University of California at Berkeley. If your knowledge of the UNIX system is sketchy or nonexistent, we recommend reading the manual entitled Beginner's Guide to the Sun System and finding your way through the various manuals by following the interesting pointers from there on.

Existing Customers Please Note
that if you already have a Sun System, these notes provide a description of the major features of this release and changes since the previous release.

This document is in three major parts:

Part I contains an overview of what the Sun System offers over and above the previous release in terms of hardware and software support.

Part II contains a description of changes to the system between Sun System Release 1.1 (May 1984) and Sun System Release 2.0 (May 1985).

Part III is a summary of the differences between the Sun System and the Berkeley 4.2 Software Distribution. Sun has embarked upon a long-term program to improve upon the system. To this end, there are inevitable discrepancies between the Berkeley Software Distribution and the Sun version of this software. Part III describes those differences that are likely to have substantial impact.

In Parts I and II of this document, we cover three topics of interest to you:

New or Changed Utilities
describes how the utilities (user-accessible programs) have changed in terms of new utilities, changes to the user interface, or utilities that have been deleted.

---

‡ Sun Workstation, and the combination of Sun with a numeric suffix are trademarks of Sun Microsystems, Inc.
† UNIX is a trademark of Bell Laboratories.

New or Changed System Interfaces
describes changes to kernel calls and library routines which necessitate changes to application code or recompilation.

New or Changed Manuals
describes new or changed documentation.

Part I — Overview of Sun Release 2.0

This part of the document contains an overview of the Sun System hardware and software offering for Sun System Release 2.0 (May 1875).

1. New Hardware Products

1.1. Sun-2 Model 50 Desktop and Model 160 Color Deskside Workstations

The Sun-2 Model 50 Desktop Workstation is based on the Model 50 single-board CPU. The model 50 is a diskless, standalone workstation.

The Sun-2 Model 160 Color Deskside Workstation is based on the Model 50 single-board CPU and the Sun-2 high-resolution (1152 × 900 × 8 bits/pixel) color board. The system runs non-interlaced with a 60 Hz refresh rate on a 19-inch color monitor. Various expansion boards and disk options are available.

The most significant departure from previous Sun standards is the use of the VME bus for the Model 160 boards. Using the VME bus has an impact on users wishing to write device drivers for the system.

Both the Model 50 and the Model 160 have onboard Ethernet controller hardware.

2. New Software Products

2.1. Network File System and the Yellow Pages

Sun System Release 2.0 contains support for the Network File System (NFS) — a set of mechanisms for sharing files in an environment consisting of other machines, other operating systems, and other networks. Sharing is accomplished by mounting a remote filesystem, then accessing files in place. The NFS is open-ended, and Sun Microsystems encourages customers and other system vendors to take advantage of the interface to extend the capabilities of other systems.

Note that the NFS places emphasis on ***network services***, and one of the first of many network services is the ***Yellow Pages***, described below.

There are new commands to administer the NFS — these commands appear in the administration manual pages of the *Commands Reference Manual*.

The Yellow Pages (YP) is a network service to ease the job of administering the network. The Yellow Pages is a centralized read-only database. For a client on the network file system, this means that an application's access to data served by the YP is independent of the relative locations of the client and the server.

2.2. Remote Procedure Call Facility

The Remote Procedure Call (RPC) facilities provides a mechanism whereby one process (the *caller* process) can have another process (the *server* process) execute a procedure call, just as if the caller process had executed the procedure call in its own address space (as in the local model of a procedure call). Because the caller and the server are now two separate processes, they no longer have to live in the same physical machine.

The RPC mechanism is implemented as a library of procedures, plus a specification for portable data transmission, known as the eXternal Data Representation (XDR). Both the RPC code and the XDR are portable. They provide a 'standard I/O Library' for interprocess communication. Thus programmers now have a standardized access to networking without having to be concerned about the low-level details of the `accept`, `bind`, and `select` procedures. See *Networking on the Sun Workstation* for documentation on the RPC and XDR mechanisms.

2.3. SunCGI — Computer Graphics Interface (CGI) Package

SunCGI is an implementation of the March 1984 draft of the ANSI CGI standard. CGI is useful for doing fast, relatively easy-to-use 2-D graphics. CGI is simpler and faster than **SunCore** with a richer set of primitives (except for the absence of 3-D primitives). CGI provides similar performance to *pixwins* without the setup overhead of SunWindows.

2.4. New Programming Language Support

The symbolic debug utility `dbx`(1) now supports Pascal as well as C and FORTRAN77.

There is a window- and mouse-based interface to `dbx`, called `dbxtool`.

A new utility called `tcov`(1) is available for instrumenting C programs to see what parts of a program are executed by given test cases. The `tcov` utility generates an annotated listing of the source code showing how many times each basic block was executed.

2.5. New Utilities

2.5.1. Fonttool — A Font Editor

`fonttool` is a new font editor for ***vfont*** format fixed-width fonts, where characters are no taller than 25 pixels (larger characters won't fit completely onto the screen). Some of the variable-width fonts can also be edited, or at least viewed. For documentation on `fonttool`, see `fonttool`(1) in the ***Commands Reference Manual***.

2.5.2. Tektool — Tektronix 4014 Emulator

`Tektool` emulates a Tektronix 4014 terminal with the enhanced graphic module. It does this in much the same way as `shelltool` (see `suntools`(1)) emulates a regular glass tty. For documentation on `tektool`, see `tektool`(1) in the ***Commands Reference Manual***.

Part II: Changes From Release 1.1 to Release 2.0

This part of the document contains a description of changes to the system between Sun System Release 1.1 (May 1984) and Sun System Release 2.0 (December 1984).

3. Changes to the Distribution Tape

The distribution tape has been drastically reorganized. See the manual entitled ***Installing UNIX on the Sun Workstation*** — the section ***What is on the Distribution Tape?*** for specific details of the new order.

Briefly, the software needed to load the UNIX system is contained either on two half-inch (nine-track) magnetic tapes, or on three quarter-inch cartridge tapes.

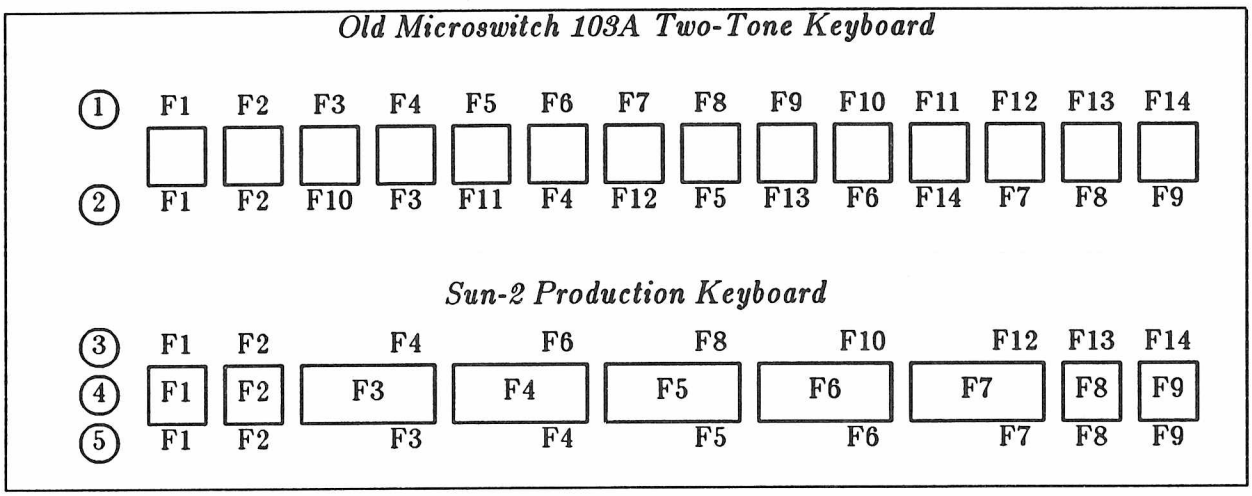
4. Changes to the Keyboard Interface

The escape sequences emitted by some of the function keys on the Sun-2 keyboard will change in 2.0 — this section provides the details. These changes should only affect people who experimented to determine what codes were generated and the wrote software to depend on those codes — such software will need changing to work correctly with release 2.0.

The function keys do not correspond to any particular ASCII characters. Therefore, following the ANSI standard, they generate strings of characters of the form <ESC> [*nnn*z, where <ESC> is the character whose decimal code is 27, and *nnn* is a sequence of numeric characters which identifies the particular key. These escape sequences were originally chosen on the basis of a prototype keyboard, which had 15 keys each in the left function pad and across the top (L1 - L15 and F1 - F15, respectively). With the double-width function keys on the existing production keyboards, not all these codes can be generated, and the correction is pretty non-intuitive. In particular, the documentation of the `mapi` and `mapo` operations in the ttysubwindow would be wrong for anyone with a production keyboard.

For Sun Release 2.0, the assignment of escape sequences reflects the product we are actually shipping. This involved changing the emitted strings. The dummy positions which never generate an output on the standard keyboard were still assigned values, so holders of the prototypes (and the old two-tone keyboards) can use all the keys they have. But these values were all assigned from codes the standard model will never generate.

The following picture summarizes the situation for the top-row function keys:



The rows in the picture are interpreted as follows:

Line 1 is the old function codes emitted by the old two-tone keyboard ***prior to Sun System Release 2.0***.

Line 2 is the new function codes emitted by the old two-tone keyboard ***after Sun System Release 2.0***.

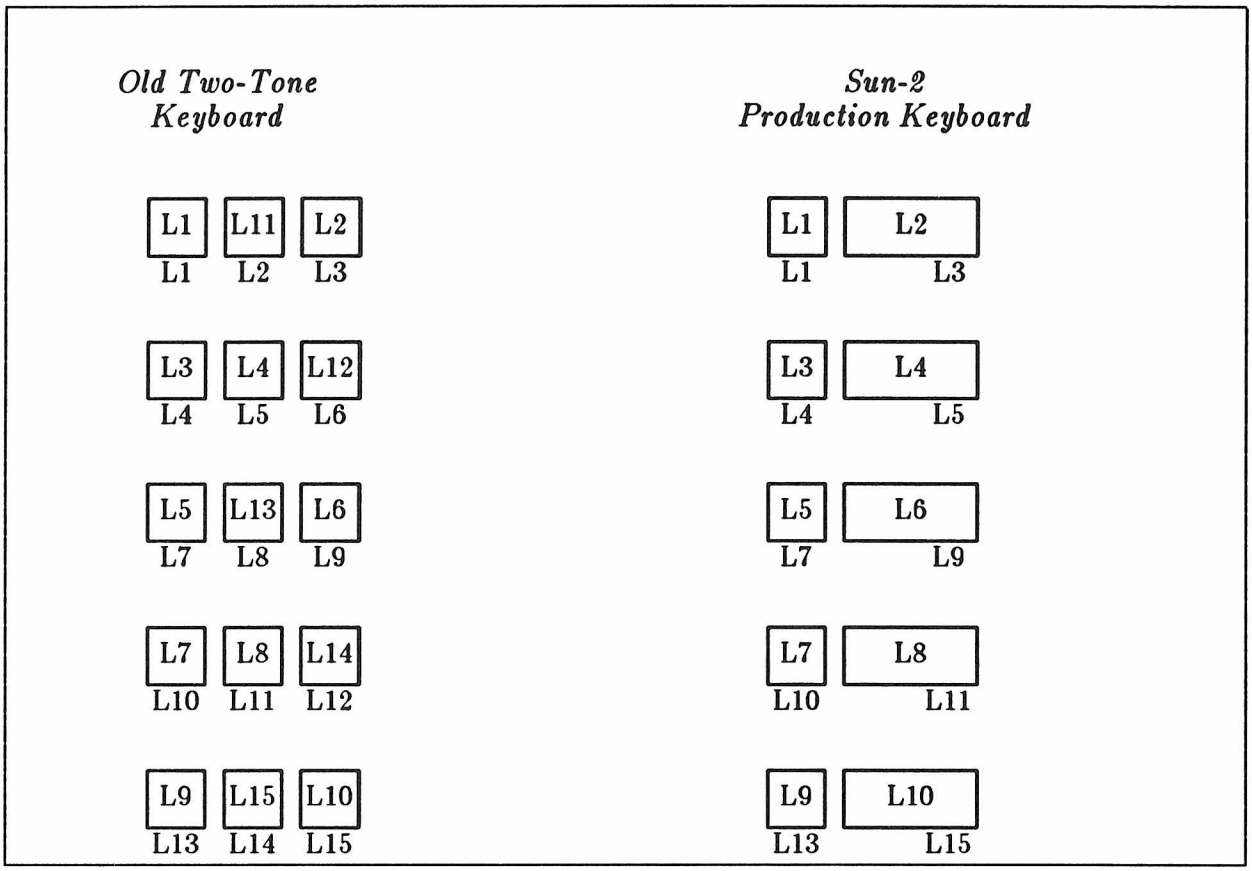
Line 3 is the old function codes emitted by the Sun-2 production keyboard ***prior to Sun System Release 2.0***.

Line 4 is the labels on the top-row function keys of the Sun-2 production keyboard. Note in particular the double-width keys.

Line 5 is the new function codes emitted by the Sun-2 production keyboard ***after Sun System Release 2.0***.

Note that a standard keyboard will not produce codes F10 and above; those switches are inactive elements, for support only, on that keyboard.

Similar changes apply on the left function pad with the extra complication that the active-inactive assignment alternates in the pair under double keys — that is, active switches are 3, 26, 51, 73, 97; R, L, R, L, R as you go down the column. The picture of the correspondence between the two keyboards looks like this:



To read this picture, please note that the numbers ***inside*** the function keys represent the new function codes emitted ***after Sun System Release 2.0***. The numbers ***below*** the function keys represent the old emitted function codes ***prior to Sun System Release 2.0***.

4.1. Changes from Revision N to Revision P PROMS

Revision P PROMs are available for Sun Release 2.0. The sections below contain details of the changes.

Device Support

The new PROMs provide support for the following devices:

- Model 50/160 CPU board including onboard Ethernet.
- Sun-2 (160) color board.
- Xylogics 472 tape controller (xt).
- 1024x1024 video (for OEMs) is supported in the same PROMs as 1152x900 video; the PROMs determine which at runtime by reading a jumper on the video board.

The "Ethernet jammed" message was changed to "Ethernet cable problem". We take more care to avoid the bugs in the Intel Ethernet chips on the VME CPU board and the Sun-2 Ethernet board.

All bootable devices are probed at power-up and the names of those found are printed in the power-up banner.

Each of the four PROM versions (50/160, 120/170, vt100, two-tone) supports a different set of devices, based on the configurations Sun sells. All devices could not be supported in all versions due to lack of space. Here's the entire list of supported PROM:

PROM LABEL	*Video Frame Buffer*	*Keyboard*	*Peripherals*
Rev P 50/160	All VME fb's	Sun-2	ie, mt, sd, st, xt, xy
Rev P 120/170	Sun-2 incl 1024	Sun-2	ec, ie, ip, mt, sd, st, xt, xy
Rev P vt100	Sun-1	VT100	ar, ec, ie, ip, mt, xt, xy
Rev P two-tone	Sun-1	two-tone	ar, ec, ie, ip, mt, xt, xy
Rev N vt100	Sun-1	VT100	ar, ec, ie, ip, mt, sd, st, xy
Rev N two-tone	Sun-1	two-tone	ar, ec, ie, ip, mt, sd, st, xy
Rev N 120/170	Sun-2	Sun-2	ar, ec, ie, ip, mt, sd, st, xy

Note that the UNIX system Release 2.0 will not support Revision L and M PROMs for diskless machines, since these PROMs do not use the Ethernet address from the CPU's ID PROM. A special fix in the Sun System Release 1.*n* allowed these machines to boot by forcing UNIX to ignore the ID PROM too.

Serial Ports

All serial ports, including keyboard and mouse, are reset better. The ports could get into states which Revision N and previous PROMs could not reset them out of. We also take better care to avoid violating the bus timing requirements of the serial chips.

Memory Usage

The "user interrupt vectors" in low memory (between `0x100` and `0x400`) are not used by the PROM Monitor for globals any more. This was necessary on VME products to support vectored interrupts, but was implemented on Multibus Revision P PROMs too.

5. Changes to the Operating System Software

In Sun System Release 2.0, the */sys* directory has moved to */usr/sys*.

5.1. Default Block Size Changed

The default block size for a file system has changed from 4K bytes to 8K bytes. The fragment size remains the same. The `setup` and `newfs` programs take this change into account when constructing *nd* partitions.

5.2. Accessing the Frame Buffer

In old versions of the system you could access the frame buffer by opening */dev/console* and then using *mmap*(2) to access the frame buffer. You can no longer do this. You must now use */dev/fb* instead of */dev/console*.

5.3. Changes to Device Driver Structures

The definition for the `mb_driver` structure found in */sys/sundev/mbvar.h* has changed. The `mdr_ioaddr` and `mdr_maddr` fields have disappeared. Customers who wrote their own device driver will have to delete these obsolete fields before compiling the driver with a 2.0 kernel.

5.4. Support for 6250 bpi Tape

Sun System Release 2.0 includes device driver support for a 6250bpi nine-track tape using a Xylogics 472 controller.

5.5. Vectored Interrupts and Changes to Config

With the use of the VMEbus, the kernel can now support vectored interrupts. Some details of the vectored interrupt support appear in *Writing Device Drivers for the Sun Workstation*.

The `config`(8) utility has had changes made to support the vectored interrupts, and sundry other cleanups. Information on the changes can be found in *Installing UNIX on the Sun Workstation* and in the `config`(8) manual page.

5.6. Changes to the `savecore` Facility

The invocation of the `/usr/etc/savecore` program in the */etc/rc.local* script has been *commented out* in Sun System Release 2.0. This means that `savecore` will no longer produce crash dumps in the event of kernel crashes. Users wishing to enable the `savecore` should remove the comment character from the line in the */etc/rc.local* file.

Note that `savecore`'s directory to save the crash dumps used to be */usr/crash*. The directory has *changed* to be */usr2/crash/hostname*, where *hostname* is the name of your machine. In order to get `savecore` to save the crash dumps, the indicated directory *must* exist.

6. Applications Software Changes in Release 2.0

6.1. Hayes Support for tip and uucp

Tip and *uucp* now support the Hayes SmartModem 1200. Many modems are compatible with the Hayes, such as the Ventel EC1200-31 and EC1200-32 (aka 1200PLUS). Such modems often

claim to be "AT" compatible or use the "AT" command set.

Switch Settings

The proper switch settings for the Hayes SmartModem 1200 for use with ***tip*** and ***uucp*** are listed below. For other Hayes compatible modems the switches should be set similarly — see your modem reference manual.

Switch	*Setting*	*Description*
1	UP	DO NOT force DTR true
2	DOWN	respond with digit result codes, NOT English words
3	DOWN	result codes WILL be sent
4	DOWN	DO NOT echo characters in command state
5	UP	DO answer incoming calls
6	UP	DO NOT force Carrier Detect true
7	UP	for connection to RJ11 modular jack
8	DOWN	enable command recognition

Tip Support

To use a Hayes modem with ***tip***, the modem type ("at" attribute in ***/etc/remote*** file) should be specified as "hayes" or "at". The phone number ("pn" attribute) can contain any valid dial commands, see your modem manual for details. The most common dial commands are:

0-9 dial that number

, pause for 2 seconds to wait for secondary dial tone

P switch to pulse dialing

T switch to tone dialing

By default ***tip*** uses tone dialing. Start the phone number with "P" to use pulse dialing.

To allow parameters of the dialer to be set, start the phone number with an 'S' to start a "Set" command. For example, to change the dial tone wait time, specify the phone number as:

```
:pn=S6=10DT4085551212:
```

Note that the Dial command must be specified explicitly.

NOTE: ***Tip*** can cycle through a set of phone numbers, trying each one until a connection is made. Previously it was common to separate the phone numbers with a comma, although this feature was never documented. Now, since comma is a valid dial character, phone numbers must be separated with a '|'. For example:

```
:pn=4085551212|4085551213|4085551214:
```

This is a potential compatibility problem and even affects people who are not using Hayes modems. They may have to change their ***/etc/remote*** file.

Uucp Support

To use a Hayes (or AT) compatible modem with ***uucp*** the modem type should be specified as ACUHAYES (or ACUAT). As with ***tip***, the phone number can contain any valid dial commands and ***uucp*** defaults to tone dialing. However, ***uucp*** uses any alphabetic prefix of a phone number

to look up a translation in the *L-devices* file. Therefore, to start a phone number with a letter insert a '–' before the letter. For example, to switch to pulse dialing, specify the phone number as

```
-P4085551212
```

An *L.sys* line that uses a Hayes modem might look like:

```
sun Any ACUHAYES 1200 -P9,4085551212 login:-EOT-login: uucp ssword: whatever
```

To use the default of tone dialing simply omit the "–P".

As with *tip*, *uucp* also allows `Set` commands in the phone number. Start the phone number with "–S" and don't forget to explicitly specify the "D" dial command.

7. Programming Language Support in Release 2.0

The notes here apply to all languages. Changes to particular languages are detailed in the sections following.

Certain constructs generate better code. In particular, integer division and modulo by positive powers of two are done in line.

There is an entirely new peephole optimizer, resulting in 2%-5% smaller code size. It also repairs many longstanding optimization bugs.

7.1. Changes to the Assembler (as)

Much faster for larger programs.

New options to control span-dependent instruction resolution:

–J make all branch instructions long

–j make all branch instructions medium length

–h make all call instructions long, all other branches medium length

These all speed up assembly further for large programs.

Identifiers can now be 512 characters long, rather than only 50; the main impact of this is to allow deeper nesting of procedures in Pascal

7.2. Changes to the Linker (ld)

The **–s** option now works.

New options added:

–Ttext *hex*
Specify the origin of the *text* segment — same as the existing **–T** option.

–Tdata *hex*
Specify the origin of the *data* segment.

These options are of interest only to PROM programmers or cross-development programmers. Programs linked with them cannot be run under the UNIX system.

7.3. Changes to the Librarian (ar, ranlib)

Ar's name truncation now applies uniformly. This had been a bug which caused a new version of an archive element to be added at the end of the archive, rather than replacing an older version, if the name was > 14 characters long.

Ranlib's arbitrary restrictions on the number of names has been removed. This was a problem for users building large libraries.

7.4. Changes to the C library (/lib/libc.a /usr/lib/libc_p.a)

A program which does not exit via an `exit` statement, but just 'falls off the end', normally returns to the system with a return value (exit status) corresponding to the last expression computed. To ward off potential future problems, the C library has been changed so that a program that 'falls off the end' always returns a 0 status code.

`atof()` and `ecvt()` are much improved in both speed and accuracy. They no longer use floating-point arithmetic.

`bcopy()` and `bzero()` can now operate with counts larger than 65536 — the new limit is 2^{31-1}.

Negative results of integer divisions when the program was linked with the **-p** or **-pg** option were erroroneously positive. This has been repaired.

`on_exit`(3) — new library call added so that programs can name their own termination handlers. See the `onexit`(3) reference manual page for details.

Memory management has been changed in the C library: `malloc()` and `free()` have been rewritten for much better performance for programs that allocate large numbers of blocks.

Blocks allocated by `valloc()` can now be given to the `free()` function.

A new `memalign()` function allocates storage on a specified alignment boundary.

Heap checking is done in programs linked with a debugging version of the ***malloc.o*** object file. Two new entry points (only in the debugging version) are:

`malloc_debug()` set the level of error diagnosis.

`malloc_verify()` scan the entire heap for inconsistencies.

7.5. Changes to the C Programming Language

Error Messages

Syntax error messages have been enhanced to give more information about the location of the problem.

Several bugs were fixed in semantic error recovery routines to permit compilation to continue where otherwise the compiler would dump core.

The *cc* 'driver' program, */bin/cc*, now interprets the status returned by the various compiler passes, rather than simply reporting 'compiler error' when one of them dumps core.

The C preprocessor has been modified to produce more comprehensible error messages.

Error detection

The function declaration

```
extern int function_name(param);
```

is now correctly tagged as incorrect.

The sequence

```
<any statement>                 /* note absence of a semicolon */
asm("something");
```

is now correctly tagged as incorrect, whereas previously this sequence would compile correctly but sometimes generate incorrect code.

The procedure definition

```
int f(x)
static x;
{
    return x ;
}
```

is now correctly tagged as incorrect.

7.5.1. Significant Bug Fixes in the C Programming Language

- The optimizer option **−O** may now be used in conjunction with the **−fsky** option.
- Pointers to functions of type `void` are now permissible.
- Huge `struct` assignments work.
- `Unsigned char` and `unsigned short` now work as parameter or function result type.
- `register float` variables are no longer a problem.
- Conversions of very large floating point constants in the source are now correct. This was caused by a bug in the `atof`() library routine used in the compiler.
- Comparisons of floating point versus integer expressions were done (incorrectly) as integer comparisons.

7.5.2. Miscellaneous Improvements in the C Programming Language

Code for large, sparse `switch` statements is more compact.

Returning from `main()` now always results in an exit code of zero. Use `exit()` to exit otherwise.

New *cc* option **−v** makes *cc* show the commands it issues to run the separate compiler phases.

New *cc* option **−a** to generate test coverage analysis information for later analysis by the *tcov*(1) utility.

7.5.3. Code Coverage Tools

There is now support for code coverage in the shape of a new option to the C compiler to use another preprocessor, plus a utility to generate statistics about the code.

Using the **−a** option on the C compiler command line runs a new preprocessor — ***/usr/lib/bb_count*** — over your program after the C preprocessor — ***/lib/cpp*** — and before the compiler gets the code. The new preprocessor inserts code into the program to count how many times each basic block in the program is executed. After the program is compiled with this option, there will be a ***.d*** file for every ***.c*** file you mentioned. The ***.d*** accumulates execution data for the corresponding source file. A new program called ***tcov*** can then be run on the source file to produce a ***name.tcov*** file which is an annotated listing of the source with the execution counts included.

This process has two goals:

1. A test coverage tool to tell you what percentage of the your code has been executed.
2. A performance analysis tool.

The **−a** option must be used on the load line as well as the compile line.

See the new *cc*(1) manual page for details of the **−a** option and the *tcov*(1) manual page for details of the report generating program.

7.6. Changes in the FORTRAN Programming Language

Changes in the FORTRAN Compiler

The peephole optimizer is now applied to FORTRAN programs when **−O** option specified.

FORMAT statements are now checked for validity at compile time. It used to be possible for a program to compile and run if it contained an incorrect but unused FORMAT statement. This is no longer so.

Changes in the FORTRAN I/O library

Extensive work has been done on the FORTRAN I/O library: FORMATTED I/O, especially numeric conversion, is noticeably faster than before.

Numeric conversion on real numbers is much more exact than before.

A FORTRAN program can now read in IEEE indeterminate forms using FORMATTED or list-directed input: it will accept `Inf` for single- or double-precision infinity, and `NaN` or `NaN(`***<hexadecimal constant>***`)` for Not-a-Number. `NaN` is equivalent to `NaN(1)`. These forms may be preceeded by an optional sign. These are the representations used on output, also.

Magnetic tape files can now be connected for sequential, formatted or unformatted I/O. Some restrictions still apply: formatted records cannot be written that are larger than the tape block size; there is no way to deal sensibly with multiple files on a single tape. The default buffer size

for tape is 64*1024 bytes. See below on how to explicitly control this.

There is a new parameter to the OPEN statement called FILEOPT, which takes a string expression. There are three possible subparameters:

BUFFER=<*decimal constant*>	Determine size of I/O buffer. Useful mainly for writing tape.
NOPAD	Suppress infinite blank padding of records on formatted reads. By default, padding normally is done.
EOF	Open file at end rather that at beginning.

For instance:

```
OPEN( 7, FILE='/dev/rmt0', FILEOPT='BUFFER=10240, NOPAD')
```

The subparameter keywords may appear in either upper or lower case.

7.7. Changes to the Pascal Programming Language

With the advent of Sun System Release 2.0, `pc`, the Sun Pascal compiler is now well integrated with our other supported languages and with the `dbx` symbolic debugger. In addition, the Pascal compiler's performance has been significantly improved over previous versions. Consequently Sun is moving toward gradually desupporting the Pascal interpreter `pi`. As the first stage of this process, we are including in the current release of `pc` (the Pascal compiler) several new extensions to the Pascal language which are not supported by `pi` (the Pascal interpreter). The second stage will be accomplished in the next major software release: it will include no bug fixes to `pi`.

7.7.1. New Features and Improvements to the Pascal Programming Language

Further details on the following items may be obtained in the Sun Pascal User's Manual, or in other documentation as indicated.

Note: some items, marked (pc only), are supported only by *pc* (the Pascal *compiler*), and are not supported by *pi* (the Pascal *interpreter*).

Symbolic Debugging now Supported (pc only)

The Pascal compiler *pc* and the symbolic debugger *dbx* now work together. *Dbx* supports Pascal programs with functionality comparable to that for C and FORTRAN, but adapted to Pascal data types, scoping, and other language-specific constructs. See the documentation on *dbx*(1) for more details.

Conformant Array Parameters

Conformant array parameters are now supported as required by the ISO Pascal Standard. This feature allows a procedure or function to handle arrays of different dimensions as arguments.

`otherwise` *clause in* `case` *statement*

`case` statements may specify a default action or **`otherwise`** clause. The default action is specified by a statement preceded by the keyword **`otherwise`**; this must be the *last* statement in the **`case`**-statement list. The default action is executed if the **`case`** selector does not match

any of the specified **case** label values. Without the **otherwise** clause, this situation would result in a run-time error and termination of the program.

`shortreal` *and* `longreal` *Types (pc only)*

The Pascal compiler *pc* now supports both single- and double-precision floating point. Single- and double-precision floating point types are denoted by the names **shortreal** and **longreal**, respectively. The standard type **real** implements double-precision floating point, as in previous versions of Sun Pascal.

> Values of type **shortreal** are identical to **float** operands in C. However, whereas C converts value parameters of type **float** to **double**, Pascal does not do the corresponding conversion.

External FORTRAN and C declarations (pc only)

The **external** directive for **procedure** and **function** declarations is extended to allow the optional specification of the source language of a separately compiled procedure or function. The directives **external fortran** and **external C** direct pc to generate calling sequences compatible with FORTRAN 77 and C, respectively. The resulting changes in the calling sequence are as follows:

external fortran:

1. For value parameters, the compiler creates a temporary copy of the actual argument's value in the caller's environment, and a pointer to the temporary is passed on the stack. This obviates the need for the user to create otherwise useless temporary variables.
2. The compiler appends an underscore (_) to the name of the external procedure, to conform to a naming convention of the F77 compiler. Note that names of Pascal procedures called *from* Fortran must supply their own trailing (_). This may be done using a #define preprocessor declaration to minimize impact on the rest of the program.

Note: multidimensional Pascal arrays are not compatible with Fortran arrays; since FORTRAN uses column-major ordering, a multidimensional Pascal array passed to FORTRAN will appear to have been transposed.

external C:

1. Value parameters of type **shortreal** are converted to type **longreal**.

Improved Code Quality and Compilation Speed (pc only)

Generated code is up to 40% faster and about 40% smaller than that produced by *pc* in Sun System Release 1.1. Improvements include constant expression evaluation, inline evaluation of some library routines, and other local optimizations. Particular improvements have been made in runtime range checking by eliminating unnecessary library call overhead, and by removing range checks involving for loop variables from the loop whenever feasible.

Compilation speed is also improved by 30-40%. Much of this improvement is due to changes in the assembler and code generator, though some is brought about by reducing the amount of code produced by the Pascal frontend.

sizeof *operator*

The **sizeof** operator returns the size of a specified type or variable. If the size of a variant record type is to be computed, an optional list of variant tag values may be used to specify a particular variant of the given record type, as for the standard procedures **new** and **dispose**.

sizeof() is a compile-time function and so does not generate any code, other than the constant value it returns.

Since **sizeof**() is a compile-time function, an attempt to apply it to a conformant array parameter is treated as an error by the compiler.

Note that **sizeof** is now a reserved word, unless the **-s** option (compile only standard Pascal) is in effect.

Bit Operations on Integral Types (pc only)

Pc now provides access to the same bit operations available in C. The operations are specified as calls to the following predefined functions:

```
x := land(y,z);     bitwise AND
x := lor(y,z);      inclusive OR
x := xor(y,z);      exclusive OR
x := lnot(y);       bitwise NOT
x := lsl(y,z);      logical shift left
x := lsr(y,z);      logical shift right
x := asl(y,z);      arithmetic shift left
x := asr(y,z);      arithmetic shift right
```

Each function takes one or two arguments of **integer** type and returns a result whose type is the wider of the two operand types. The result is computed in-line, producing faster and smaller code than an equivalent external function call.

Preprocessor Facilities (pc only)

Preprocessor facilities (conditional compilation, macros) may be used as in C. Caution: comments containing # in column 1 are interpreted by the preprocessor, and are considered wrong if not recognized as one of the valid preprocessor directives (**#ifdef**, **#if**, **#else**, **#endif**, and so on). Also, whereas Sun C reserves the symbols **sun**, **unix**, and **mc68000**, they are not reserved for Pascal.

Underscores Allowed in Identifiers

The syntax of Pascal identifiers has been extended to allow underscores (_) in all character positions except the first. This allows access to more Sun library routines than in earlier versions.

Note that leading underscores are not supported. Objects with names beginning with _ must still be accessed indirectly by going through a C routine.

Version Identification (pc only)

The version of **pc** used to compile a given object file may be identified by the command

```
tutorial% nm -ap <file> | grep PC | head
```

The first line containing the string "PC" indicates the version of the compiler used, and the date of its generation, for example,

```
tutorial% nm -ap a.out | grep PC | head
00000000 - 00 000d    PC 3.4 (10/4/84)
```

New Command Line Options for Pascal System Programs

–L Map characters in identifiers to lower case (also in **pxp**).

–P partial evaluation of boolean operators (**pc** only)

–I*dir* alternate include pathname

–D*name*=*def*

–D*name* define preprocessor symbol (**pc** only)

–U*name* delete definition of preprocessor symbol (**pc** only)

–fsky Generate in-line SKY code for floating point (**pc** only)

7.7.2. Bugs Fixed in Pascal Programming Language

Multidimensional Array Declarations

Arrays of arrays and multidimensional arrays are treated equivalently, as specified in the standard. For example,

```
array[1..10] of array[1..6] of real;
```

and

```
array[1..10,1..6] of real;
```

are treated as equivalent, as are `a[i][j]` and `a[i,j]`. In versions of **pc** and **pi** before Sun System Release 2.0, they were not, and substituting one for the other caused a compiler error message.

Procedures and Functions as Parameters (pc only)

The implementation of procedures and functions as parameters has been changed to be compatible with FORTRAN 77 and C, and to provide a more efficient calling sequence. Outer block Pascal procedures and functions may be passed to C and FORTRAN 77, and vice versa.

Note: ***nested*** Pascal procedures and functions should not be passed as parameters to external C procedures or FORTRAN 77 routines. **Pc** will issue a warning if this is attempted.

Note: external C or FORTRAN 77 routines which require nonstandard calling sequences (for example, FORTRAN 77 value parameters) should not be passed as parameters to routines of any language. **Pc** will issue a warning if this is attempted.

The major impact is that any C wrappers that were previously written to be cognizant of the Pascal calling sequence will have to be changed. In many cases, they can be discarded because many wrappers were written to turn Pascal calling sequences into C calling sequences.

Returning Program Status

A change in the startup code for Pascal ensures that if the C library routine ***exit***(3) is called, all pending Pascal (and FORTRAN) I/O will be completed and all files closed. In particular, ***exit***() may be called from anywhere to return the program status to the calling entity (the shell, make, etc.)

Accuracy of Read(real)

`read(real_number)` now handles zeros immediately to the right of the radix point correctly. Formerly this resulted in a loss of accuracy.

Long Names and/or Nested Procedure Declarations

Long names are now handled correctly. Names longer than about 30 characters used to abort compilations in the assembly phase. This was particularly annoying when the long names came about as a result of nested procedure declarations (for which inner names are qualified by chaining together the names of the enclosing procedures)

pxp Bug Fix

Pxp now formats routines with procedures or functions as parameters correctly. Previously, nested parameter lists were omitted, as in the obsolete Jensen & Wirth definition of Pascal.

Known Bugs in the Pascal Programming Language

1. Operands of binary set operators {*, +, -} are required to have identical types. The standard permits different types, as long as the base types are compatible.
2. According to the standard, the expression `[maxint..-maxint]` should be equivalent to `[]`. *Pc* and *pi* will both refuse to evaluate sets with elements larger than indicated by the definition of type 'intset' (predefined as `set of 0..127`).
3. `eof(`*some_file*`)` is not true after reading the last character of the textfile *some_file*.
4. The value of m **mod** n is not computed correctly for negative values of m. According to the standard, the divisor must be positive and the result is negative. The correct result may be obtained by:

```
result := m mod n;
if result < 0 then
    result := result + n;
```

Missing Functionality

1. There is no way to recover from an error when resetting or rewriting a file. For example,

```
reset(f,'any_file')
```

will abort if the file *any_file* does not exist in the current directory.

2. There is no clean way to extract the `(FILE*)` from a Pascal **file** variable in order to do C standard I/O library calls on it.
3. There is no way to disable type checking when calling C routines with variable numbers of arguments or with unchecked types (such as: *printf()*, *fread()*, *fwrite()*, *malloc()*, ...).
4. There is still no decent support for strings. However, the string functions in the C library may be sufficient for some applications.

7.8. Release 2.0 Floating Point and Elementary Functions

The software for floating point arithmetic has been revised for Sun System Release 2.0 to make it closer to the IEEE standard. The software for elementary functions has been revised to improve accuracy and speed. For SKY board users, Sun System Release 2.0 includes SKY Computers' microcode release 4.00 which mends some previous defects. Sun System Release 2.0 also gives FORTRAN and Pascal programs faster access to the single-precision square root and elementary functions available on the SKY board.

7.8.1. Software Floating Point and IEEE Standard

Release 2.0 floating point software meets more of the specifications of the proposed IEEE standard P754 for binary floating point arithmetic than the previous release. Release 2.0 floating point software is contained in libc, except for square root, which is contained in libm. Thus this software will normally be linked with programs written in C, FORTRAN, or Pascal.

The following paragraphs compare the requirements and recommendations of the IEEE standard with the Release 2.0 floating point software:

Floating Point Formats

The single-precision storage format is required by the IEEE standard, and the double-precision and double extended storage formats are optional. All formats include special representations for zero, subnormal numbers, infinity, and not-a-number (NaN). Release 2.0 floating point software provides single- and double-precision including the special representations.

Rounding

Rounding to nearest is the default rounding method; the standard also requires three other directed rounding methods. Release 2.0 provides only the default rounding to nearest.

Operations

Release 2.0 provides the following operations specified in the standard: addition, subtraction, multiplication, division, square root, conversion between single- and double-precision floating point formats, conversion from integer formats to floating point formats, and conversion between floating point and ASCII. Release 2.0 does not provide the remainder operation specified in the standard. The floating point comparison operators "equal" and "not equal" conform to the standard for all operands, but the operators "less than," "greater than," "less than or equal, "greater than or equal" do not conform to the standard if one or both operands is a NaN. Release 2.0 does not provide the operation of rounding a floating point number to the nearest integer in floating point format, but the generic FORTRAN function AINT provides the operation of rounding toward zero a floating point number to an integer in floating point format. Release 2.0 does not provide the operation of rounding a floating point number to the nearest integer in integer format, but does provide the operation of rounding a floating point number to the nearest integer toward zero in integer format.

Infinity, NaNs, and Zero

Release 2.0 conforms to the standard's requirements, except that signalling NaNs are not provided.

Exceptions and Traps

Release 2.0 does not signal exceptions or allow trapping on them.

SKY Board Floating Point

The SKY board provides single- and double-precision floating point storage formats according to the standard, and provides addition, subtraction, multiplication, division, square root, conversion between floating point formats and between floating and integer formats, and floating point comparison, but the details of these operations do not always match Sun's software floating-point or the IEEE standard. Note in particular the following differences:

- Conversion from single-precision to double-precision is erroneous for subnormal numbers, infinities, and NaN's. This also affects single-precision functions which Sun's ***libm.a*** functions compute in double-precision, including power (x^y), asin, acos, sinh, cosh, and tanh.
- Like the software floating-point comparison, the SKY floating-point comparison does not handle NaN's correctly. The software floating-point comparison always denies that $x=y$ whenever x or y is a NaN. The SKY comparisons will sometimes affirm that $x=y$ when x or y is a NaN and the other is a NaN or infinity.
- Square root of zero is erroneous. Square roots of positive numbers may differ by two units in the least significant bit from the IEEE standard.
- Other rounded results may differ by one unit in the least significant bit from the result specified in the IEEE standard.
- The sign of zero results may differ from the result specified in the IEEE standard.

Software Elementary Functions

The mathematical library file ***libm.a*** provides software to compute a number of elementary functions including the following common ones: exp, log, power (x^y), sin, cos, tan, asin, acos, atan, sinh, cosh, tanh. Release 2.0 contains revised double-precision versions of all these functions and revised single-precision versions of exp, log, sin, cos, tan, and atan. Single-precision asin, acos, sinh, cosh, and tanh are computed using the double-precision software.

The revisions were primarily intended to improve accuracy and handling of exceptional arguments, but in many cases the code is faster also.

SKY Board Elementary Functions

The single-precision ***libm.a*** functions sqrt, exp, log, power, sin, cos, tan, and atan are provided by the SKY board if it is present. Some of these functions are more accurate than others, and all differ from the software versions in their treatment of exceptional operands.

SKY Board versus Software

Programs that perform intensive floating point execute up to three times faster when a SKY board is installed. As indicated above, however, the SKY board arithmetic is not as close to the IEEE standard as the software arithmetic, and the SKY board single-precision elementary functions sometimes produce unexpected results. Users are responsible for ensuring that their programs only use the SKY board within its intended domain of applicability.

SKY/Software Switching Subroutines

By default, programs compiled without the **−fsky** switch will use the SKY board if available and software otherwise. Release 2.0 of the mathematical library ***libm.a*** contains six entry points to allow FORTRAN programmers to use both SKY and software floating point and functions in the same program:

```
call ffloat
call sfloat
call vfloat
call ffunc
call sfunc
call vfunc
```

`ffloat` causes subsequent arithmetic operations to be performed in software; `sfloat` causes them to be performed on the SKY board, which must be installed; and `vfloat` causes them to be performed on the SKY board if available and in software otherwise, which is the default. `ffunc`, `sfunc`, and `vfunc` provide analogous control for single-precision functions. The 'arithmetic operations' controlled by `ffloat`, `sfloat`, and `vfloat` include single- and double-precision addition, subtraction, multiplication, division, comparison, conversion between floating point and integer, and conversion between single- and double-precision. The 'functions' controlled by `ffunc`, `sfunc`, and `vfunc` include single-precision square root, exp, log, power, sin, cos, tan, and atan. Double-precision functions are not computed directly by the SKY board but by software routines which use the floating point controlled by `ffloat`, `sfloat`, and `vfloat`.

7.9. Changes to Symbolic Debug Utilities (`dbx` and `dbxtool`)

If the object file being debugged is changed during a debugging session, ***dbx*** will correctly reinitialize itself on the next **`run`** command.

In a stack trace, ambiguous function names are now qualified.

Floats are now printed correctly with the machine-level commands.

Known Bugs in dbx

Variables within nested blocks in C programs are handled incorrectly. *Dbx* displays the wrong locations from the stack. This bug is only likely to bite people who declare variables in compound statements and then try to displa the variables using ***dbx***.

In ***dbxtool***, the syntax of the `unbutton` command has changed: the syntax used to be

`unbutton` *seltype* *command*

the syntax is now changed to be

`unbutton` *command*

7.10. Sky Microcode Changes

Sun System Release 2.0 includes version 4.00 of the Sky microcode which fixes several problems with the microcode in Sun System Release 1.1.

The following problems present in the Sky microcode version shipped with Sun System Release 1.1 have been removed in version 4.00, shipped with Sun System Release 2.0:

Feature	*Problem Fixed*
Zero Compare to Subnormal	In single and double precision comparisons, zero compared equal to subnormal numbers.
Incorrect Compare	In double precision comparison certain numbers such as 10.0 and $10.0+1^{-13}$ compared greater than or equal.
Underflow to Zero	In single and double precision add, subtract, multiply, and divide, underflows were often set to zero even when the rounded result could have been represented as a subnormal number.
Underflow to -Infinity	Single precision exp and ** often set underflows to negative infinity.
Elementary Function Accuracy	Single precision **, exp, log, sin, cos, and tan were quite inaccurate. Many errors of several hundred units have been reduced to less than ten units.
Trigonometric Argument Reduction	Trigonometric functions of arguments whose magnitudes exceeded about 32000 were incorrect.
Incorrect Multiply	(Present in Sky versions 3.00 and 3.25, not part of any major Sun release): In double precision multiplication half the precision would occasionally be lost.

8. Changes in the 2.0 Release of the SunCore Graphics Package

This section is a summary of changes to the **SunCore** graphics package for the 2.0 Release. Please see the appropriate sections of the **SunCore** Programmer's Reference Manual for details.

8.1. Added `set_pick` Function

The size of a square pick aperture can be specified so that the application program can set the sensitivity of the PICK device.

8.2. Additional Raster Fonts

There are now six raster (STRING precision) fonts available for **SunCore**.

8.3. Support for Additional Devices

SunCore can now run on */dev/cgtwo* (Sun-2 color board) as well as all previously supported display surfaces.

8.4. Speed Improvements

Picking has been speeded up between 20% (for a display file of a few untransformed segments of very few primitives) and 80% (for a display file of many fully transformed segments of many primitives).

Transformations have been speeded up by 10% - 20%.

Other various improvements have been made in **SunCore** and the underlying libraries and compilers, so a general (5% - 10%) improvement in performance should be seen in **SunCore**.

8.5. Bug Fixes

- Calling the `text()` function with a space character as part of the text string, ***character_precision*** set to STRING, and ***font*** set to SPECIAL, previously generated a segmentation fault — this problem has been fixed.
- Calling the `set_font()` function with ***character_precision*** set to STRING, would previously use the default font the first time the function was called instead of using the requested font — this problem has been fixed.
- The actual width of fat vectors (line with ***line_width*** >0) are now the same independent of the displayed slope of the line.
- Markers are now centered at the current position.
- `set_zbuffer_cut` now reports an error if the surface does not support hidden surfaces.
- All error messages are now sent to the standard error file.
- Calling the `terminate_core()` function while there were still many retained segments took a long time — this problem has been fixed.
- Calling the `set_ndc_2()` function on a raw ***bw2*** surface no longer produces a segmentation fault when the x and y axes are not full scale. This problem also manifested itself by producing strange results from `size_raster()`.
- Every line segment created by `polyline()` can now be picked.
- `text()` no longer produces a floating-point exception when the projection is perspective.
- The machine no longer hangs after a call to `polygon_abs_3()`.
- Single pixel-high polygons no longer generate a segmentation fault.
- Horizontal edges of a polygon now create the same scan line segments independent of the directional sense of the polygon.
- The SKY version of the **SunCore** library now has a `get_view_surface()` function.

- The Pascal and FORTRAN wrappers for `get_mouse_state()` now have the correct number of arguments.
- The problem of dragged rasters becoming invisible has been fixed.
- `set_pickid()` now sets the pickid for all subsequently created primitives in a segment, not just for the next primitive.

9. Changes in the 2.0 Release of SunWindows

9.1. Reorganization of /usr/suntool Directory

The */usr/suntool* directory has been reorganized out of existence. The files from */usr/suntool* have been distributed as follows:

- `suntools`, `clocktool`, `gfxtool`, `shelltool`, `coretool`, `toolplaces`, `lockscreen`, `adjacentscreens`, `perfmon`, `perfmeter`, `fonttool`, and `tektool` have been moved to */usr/bin*.
- `bouncedemo`, `spheresdemo`, `jumpdemo`, and `framedemo` have been moved to */usr/demo*.
- The directory *globeframes* has been moved to */usr/demo/globeframes*.
- The directory *tekfonts* has been moved to */usr/lib/fonts/tekfonts* and the directory *fixedwidthfonts* has been moved to */usr/lib/fonts/fixedwidthfonts*.
- The directory */usr/include/images* contains the icons and images used by the window system.
- The source for the window system that comes as part of the binary release is in */usr/src/sun/suntool*. The source that accompanies the SunWindows Tutorial is located in the */usr/src/sun/suntool/tutorial* directory.
- The default *rootmenu* has been moved to */usr/lib/rootmenu*.
- The object and source for `optiontool`, `panetool`, `mousetool`, and selections are not part of the Sun System Release 2.0.

9.2. Changes to the User Level Software

Incompatibilities

Shelltool takes **−C** instead of `CONSOLE` to make it the console.

Shelltool and *gfxtool*
now need `csh` in front of the **−c** argument. Csh used to be implied.

Suntools Program Extensions

The syntax and semantics of the *.suntools* have been extended:

- Comment lines may be included by having a "#" as the first character.

- Command lines may be placed in *.suntools*. No interpretation (*a la* shell) of the command line is done. Combined with generic tool command line arguments, this allows tool placement, etc.

The file */usr/lib/rootmenu* contains a description of the menu displayed over the background grey. Setting a ROOTMENU parameter in your environment to another file changes the place from which the menu description is gleaned. You can customize the menu by modifying the file. Lines in the file start with the string to display in the menu followed by a command. Strings with embedded blanks are delimited with double quotes. Comment lines start with a # sign. Here are the commands that you can supply:

EXIT Exit the suntools program, after user confirmation. For example:

```
Exit    EXIT
```

REFRESH Redraw the entire screen. For example:

```
ReDisplay       REFRESH
```

MENU Stack a menu on the pile with the Suntools menu. Get the menu contents from the filename that follows. Name the menu with the string preceding MENU. For example:

```
"More Tools"    MENU    /usr/tmp/rootmenutool
```

Lines without recognized commands are treated as command lines and executed. For example:

```
Shell           shelltool
Lock            lockscreen -r
```

No interpretation (*a la* shell) of the command line is done.

Ttysubwindow Changes

The selection mechanism has been extended. Here is the meaning of the mouse buttons:

Mouse Button	*Meaning*
left-down	Start selection. Moving mouse while button down changes selection.
left-up	End selection.
middle-down	Start adjust selection. Moving mouse while button down changes selection.
middle-up	End adjust selection.
right	Bring up the menu.
shift-right	Accelerator for the Stuff menu operation that doesn't bring up the menu.

Multi-clicking on the left or middle button rotates the selection level from character to word to line to paragraph and back to character. Adjust starts at the same level select ended at. Select always resets the selection level to character. A multi-click is defined as clicking over an existing selection within 1/2 a second of releasing the same mouse button.

There is a mode, called page mode, that a tty subwindow can be in so that voluminous output (say from an `ls` command) doesn't scroll off the screen before you can see it. This saves one from having to redo a command and piping it through ***more***(1) or ***page***(1) if the original command generated too much output. When page mode has halted output to the screen a tiny stop sign is placed in the mouse cursor. Typing any key releases the screen for further output. Page mode is disabled when the window is in ***cbreak*** mode or in ***raw*** mode, so page mode does not work with ***rlogin***.

The menu has expanded:

`Stuff`	Stuff the current selection as input.
`Page Mode On`	Turn on page mode (if off).
`Page Mode Off`	Turn off page mode (if on).
`Continue`	Restart output (if stopped by page mode).

Whenever a tty subwindow is created, the startup file ***~/.ttyswrc*** is parsed for tty subwindow specific parameter. The command format of this file is:

#	Comment.
`set` ***variable***	Turn on the specified ***variable***.
`mapi` ***key text***	When ***key*** is typed pretend ***text*** was typed.
`mapo` ***key text***	When ***key*** is typed pretend ***text*** was output.

The only currently defined ***variable*** is `pagemode`.

Key is one of L1-L15, F1-F15, T1-T15, R1-R15, LEFT, RIGHT.

Text may contain escapes such as \E (escape), \n (carriage return), ^X (control-X), and so on — see the ***termcap***(5) manual page for the format of string escapes recognized.

> It is likely that the `mapi` and the `mapo` functionality will be replaced by another keymapping mechanism in the future.

It is now possible to have terminal based programs drive the tool in which its tty subwindow resides by sending it special escape sequences. These escape sequences may also be sent using the `mapo` function described above. The following functions pertain to the tool in which the tty subwindow resides, not the tty subwindow itself.

Sequence	*Description*
`\E[1t`	open.
`\E[2t`	close (become iconic).
`\E[3t`	move, with interactive feedback.
`\E[3;TOP;LEFTt`	move, to TOP LEFT (pixel coordinates)
`\E[4t`	stretch, with interactive feedback
`\E[4;WIDTH;HTt`	stretch, to WIDTH HT size (in pixels)
`\E[5t`	top (expose)
`\E[6t`	bottom (hide)
`\E[7t`	refresh
`\E[8;ROWS;COLSt`	stretch, to ROWS COLS size (in characters)
`\E[11t`	report open or iconic, sends `\E[1t` or `\E[2t`
`\E[13t`	report position, sends `\E[3;TOP;LEFTt`
`\E[14t`	report size in pixels, sends `\E[4;WIDTH;HTt`
`\E[18t`	report size in characters, sends `\E[8;ROWS;COLSt`
`\E[20t`	report icon label, sends `\E]Llabel\E\\`
`\E[21t`	report tool header, sends `\E]llabel\E\\`
`\E]l<`*text*`>\E\`	set tool header to < *text* >
`\E]I<`*file*`>\E\`	set icon to the icon contained in < *file* > (`icontool` output format)
`\E]L<`*label*`>\E\`	set icon label to < *label* >
`\E[>OPT;...h`	turn OPT on (*OPT*=1=>*pagemode*). For example: `\E[>1h`
`\E[>OPT;...k`	report OPT, sends `\E[>OPT1` or `\E[>OPTh` for each OPT
`\E[>OPT;...l`	turn OPT off (*OPT*=1=>*pagemode*). For example: `\E[>1l`

As an example of using this facility, the following aliases can be put into your *~/.cshrc* file:

```
# dynamically set the name stripe of the tool:
alias header 'echo -n "^[]l\!*^[\"'
# dynamically set the label on the icon:
alias iheader 'echo -n "^[]L\!*^[\"'
# dynamically set the image on the icon:
alias icon 'echo -n "^[]I\!*^[\"'
```

New Tool Command Line Options

Clocktool takes **-s** switch to display second hand. There are a variety of other options to select the format of the clockface display — see the `clocktool`(1) manual page for details.

Shelltool takes **-C** instead of `CONSOLE` to make console.

Gfxtool takes **-C** also.

Shelltool and *gfxtool* need `csh` in front of **-c** argument.

Generic tool arguments (most tools accept most arguments):

Flag	*Long Flag*	*Arguments*	*NOTES*
-Ww	(-width)	columns	
-Wh	(-height)	lines	
-Ws	(-size)	x y	
-Wp	(-position)	x y	
-WP	(-icon_position)	x y	
-Wl	(-label)	*"string"*	
-Wi	(-iconic)		
-Wt	(-font)	filename	
-Wf	(-foreground_color)	red green blue	0-255 (no color-full color)
-Wb	(-background_color)	red green blue	0-255 (no color-full color)
-Wg	(-set_default_color)		(apply color to subwindows too)
-Wn	(-no_name_stripe)		
-WI	(-icon_image)	filename	(for tools with non-default icons)
-WL	(-icon_label)	*"string"*	(for tools with non-default icons)
-WT	(-icon_font)	filename	(for tools with non-default icons)
-WH	(-help)		

New Tools

`fonttool` a font editor — see `fonttool`(1) for documentation.

`perfmeter`
performance monitoring tools — see `perfmeter`(1) for documentation.

`dbxtool` a window- and mouse-based interface to `dbx` (the symbolic debug utility) — see `dbxtool`(1) and `dbx`(1) for documentation.

`tektool` a window-based Tektronix 4014 emulator — see `tektool`(1) for documentation.

Icontool Changes

The format of the files that the ***icontool*** generates has changed:

The ***old*** format was a `static` array declaration with initialization.

The ***new*** format is a comment describing the icon, follwed by the values for the icon image. Most programs use the output of the ***icontool*** by simply having a `#include` of the icon file. To convert such programs from the old format to the new format, it is now necessary to add an explicit array declaration around the `#include` statement.

Fixed Width Font Changes

New fonts:

apl.r.10 useful only for apl.

screen.r.11 smaller than the built-in font.

Fonts have been renamed to conform to a uniform font naming convention. A font name is in the form ***name.tf.sz*** where:

name The family name of the font

tf Type face — either b (bold), r (roman) or i (italic)

sz The point size of the font.

The names of the fonts in */usr/lib/fonts/fixedwidthfonts* have changed in two ways:

1. The previous point sizes gave no indication of the size of the font. We have adopted the following convention for determining the point size of a font: a 'point' is equal to a pixel on Sun screen; the point size of a font is the number of pixels from the top of a `T` to the bottom of a `y`.
2. The family name of the 'gacha' and 'sail' fonts has been changed to 'screen'. This is because these fonts have been 'tuned-up' for this release and are suffiently different from the previous fonts to warrant a new name.

For upward compatibility, the old names have been linked to the new fonts. Here is a list of the new font names and new fonts:

New Font Name	*Size*	*Old Font Name*
`screen.r.7`	6 x 8	`sail.r.6`
`screen.r.11`	7 x 11	(new)
`screen.b.12`	8 x 14	`gacha.b.7`
`screen.r.12`	7 x 14	`gacha.r.7`
`screen.b.14`	9 x 16	`gacha.b.8`
`screen.r.14`	8 x 16	`gacha.r.8`
`cmr.b.14`	10 x 16	`cmr.b.8`
`cmr.r.14`	9 x 16	`cmr.r.8`
`gallant.r.19`	12 x 20	`gallant.r.10`
`apl.r.10`	8 x 10	(new)

Symbolic links from old names to the new files are in */usr/lib/fonts/fixedwidthfonts* for upwards compatibility.

9.3. Changes to the Pixrect Library

A new pixrect driver called *cg2pixrect* is available for the Sun-2 high-resolution color board. A variety of bugs have been fixed in the pixrect code. Two new pixrect functions have been added, namely, `pr_polygon_2` and `pr_traprop`, plus routines for accessing rasterfiles. Major bugs fixed include:

- Better error handling.
- Framebuffer to memory RasterOps now use the *op* argument.
- RasterOp from primary pixrect to overlapping secondary pixrect now works.
- `pr_reversevideo` has been fixed.
- `pf_textbound` has been documented.

9.4. New Panel SubWindow Package

The 'Option subwindow Package' has been superseded by a new package called the 'Panel Package'. The panel subwindow package provides new item types, better support for images, easier layout of forms and panels, and other extensions.

All new programs should use the Panel Package rather than the Option subwindow Package.

Existing programs need not be converted at this time, as the Option subwindow package remains in this release. However, the Option subwindow Package will not be present in the next major release.

9.5. Changes in the Window System Libraries

All references are to the *SunWindows Reference Manual* unless noted. Use the index to get the exact location in the manual.

Sunwindow Library Changes

Input control

- Implemented `FIONREAD` (see `tty`(5) in the *System Interface Manual*).
- Implemented asynchronous notification of input for window devices, that is, `FIOASYNC` (see `SIGIO`).
- Fixed `F_GETFL` for `fcntl`(2), that is, `FIONBIO` (see `fcntl` in the *System Interface Manual*).

Event Reporting

- Fixed input mechanism so that `LOC_STILL/LOC_WINENTER/LOC_WINEXIT` events are properly generated.
- Fixed input mechanism so that `MOVEWHILEBUTDOWN` event comes ***after*** button going down notification.

Other Fixes

- Colormap restrictions have been eased. The foreground and background of all colormap segments used to be forced to the system wide notion of same. Now, this occurs only when the specified are equal.
- Quiting out of suntools used to sometimes log a user out. The occurrence of this behavior has been drastically reduced.
- Fixed `win_insert` bug that prevented arbitrary window tree insertion among sibling windows.

Suntool Library Changes

Tool Windows

- One can now set much of the behavior of tool windows via standard command line arguments (see `tool_parse_all`).
- It is now easier to override the tiling mechanism of a tool (see `tool_layoutsubwindows`).
- The interface to tool functionallity has been extended to use an attribute mechanism (see attributes). Access and control of tool features is much enhanced.

Sun Terminal Emulation Subwindows

- There is a new terminal emulator subwindow type that knows about its tool window (see `ttytlsw_createtoolsubwindow`). This subwindow can affect the tool window based on escape sequences sent to the terminal emulator.
- The interface for programmatically driving a terminal emulator subwindow's input and output streams has been documented (see `ttysw_output`).
- An interface for programmatically extending the interpretation of escape sequences sent to the emulator subwindow has been created (see `ttysw_escape`).
- Fix escape sequence to set cursor to start of line *x* (`^[[#H`).

Menus now can take graphic images (see `menu_display`).

New routines have been added to the ***libsuntool.a*** library to aid in run-time loading of icons. See the description of `icon_load` in the ***SunWindow Programmer's Reference Manual***.

10. Utility Packages

10.1. Changes to Rasterfile Format

The following changes have been made to the rasterfile support: In the include file ***/usr/include/rasterfile.h***, the ***ras_encoding*** field has been renamed to be ***ras_length***, which is defined to be the length (in bytes) of the following image data. This field was always zero (0) in previous releases.

Several new constants have been defined for use in the ***ras_type*** and ***ras_maptype*** fields.

New support for reading and writing rasterfiles has been added to the ***libpixrect.a*** library. See the pixrect documentation for further details.

10.2. Screendump, Screenload, and Rastrepl

The ***screendump***, ***screenload***, and ***rastrepl*** utilities have been changed. Here is the nature of the changes:

screendump The **−c** option has gone.
screendump no longer tries to get a color frame buffer if it cannot get the console buffer.
A new **−t** option specifies the ***type*** of file to generate: type 0 is old format files, compatible with 1.X software; type 1 is the standard format; type 2 is run-length encoding of bytes. You should use type 65535 to experiment with private encodings.
A new **−e** option is shorthand for **−t 2**.
There is now an optional ***output-file*** where the output will be sent. Output is to the standard output if the optional ***output-file*** is not specified.

screenload A new **−p** option means wait for a newline to be entered on the standard input before exiting.

rastrepl You can now specify an optional ***input-file*** and ***output-file***. ***Rastrepl*** reads from the standard input if the optional ***input-file*** is not specified, and writes to the standard output if the optional ***output-file*** is not specified.

11. New and Changed Documentation

We are now using `pic` and device-independent `troff` to format manuals. We gain in two areas:

- We can insert simple pictures — line drawings, flowcharts, and so on — without any pasteup.
- We have access to more fonts, specifically a `typewriter font like this` and a **`bold typewriter font like this`**. This means that code listings look like code, and examples of user input and system response look more real.

In addition, we have made a determined attempt to ensure that all manuals have front matter — table of contents, lists of tables where appropriate, and lists of figures where appropriate, and back matter — index. Some of the larger manuals such as **SunCore** have ***two*** tables of contents at the front: the first table of contents is called a ***major*** table of contents or a ***supplementary*** table of contents — it lists only the chapter titles and gives an overview of the contents and structure of the book; the second table of contents is detailed and serves as a locator by topic area.

11.1. System Setup and Administration Manuals

Sun has reorganized the System Manager's Manuals for a more logical organization. Before, you would get one large manual for a given system model. This large manual covered all aspects of hardware setup, software installation, and system adminstration. The System Manager's Manual has been 'unbundled' into three chunks as follows:

- A *Hardware Installation Guide* for each model. Includes unpacking and setup instructions for the basic workstation and its peripherals (if any) and covers configuration details for the card-cage and circuit boards.
- A generic *UNIX Installation Manual* to describe how to install the UNIX operating system for the first time.
- A generic *Adminstration Manual* to describe how to run the UNIX operating system on a day-to-day basis.

Much new material has been added, specifically in the areas concerning disk servers, line-printers, modems, and disk diagnosis and repair.

11.2. Editing and Text Processing on the Sun Workstation

Changes made to this manual were:

- Added the `pic` reference manual — `pic` is really only of interest to people who can access device-independent `troff`.
- Reorganized the `troff` manual in many ways. Added much new explanatory material.

- Largely rewrote the `refer` manual.
- Added a section on the **–man** macro package.

11.3. Language Manuals

Assembler Manual

The assembler manual was previously a part of the ***Programming Tools for the Sun Workstation***, but is now a manual in its own right with part number 800-1179-01. Some minor clarifications were made, some bugs were fixed, the 68010 specific material was folded into the appropriate places in the manual, and an index was added.

Pascal Programmer's Guide

This manual has had major surgery performed.

FORTRAN Programmer's Guide

Added a new section on tape I/O in FORTRAN. Added an entire new chapter on ***Profiling and Debugging FORTRAN Programs***.

11.4. SunWindows Manuals

The major change to the SunWindows Programmer's Reference Manual is the addition of a new chapter on the Panel Subwindow Package, plus a new appendix on converting from the Option Subwindow Package to the Panel Subwindow Package. The Option Subwindow Package was moved to an appendix. A new appendix was added describing how to write pixrect drivers.

There is a completely new tutorial manual for SunWindows.

11.5. SunCore Programmer's Reference Manual

There were two major additions to the **SunCore** Programmer's Reference Manual, and a number of minor changes. The major changes are

List of Errors
: A new appendix was added containing a list of the **SunCore** error numbers and the messages associated with those numbers

C Function Reference
: A new appendix was added containing an alphabetical reference summary of the **C** function interfaces and arguments — this list added at the request of customers.

11.6. Computer Graphics Interface (CGI) Programmer's Reference Manual

The Computer Graphics Interface (CGI) Programmer's Reference Manual is available in Sun System Release 2.0. In keeping with the **SunCore** manual, the CGI manual is a ***reference manual*** and does not purport to teach concepts of graphics programming.

SunCGI uses the March 1984 ANSI draft of the CGI standard. The SunCGI manual describes the **C** and FORTRAN interface to the CGI library. The manual also explains how to use CGI in conjunction with ***pixwins***. Finally, although the SunCGI manual is not a tutorial, detailed explanations of the error messages are provided.

11.7. Programming Tools for the Sun Workstation

Removed the assembler manual and made it into a separate manual in the programming languages area.

The *Programming Tools* manual has been made into a single manual of chapters in preparation for future reorganization.

11.8. Curses Manual

Finally, the ***curses*** manual is formatted and available for readers.

11.9. Device Drivers for the Sun Workstation

The *Writing Device Drivers for the Sun Workstation* section of the *System Internals Manual* has been made into a separate self-contained volume. There is much new material on VMEbus and vectored interrupts.

11.10. Additions and Changes to the Reference Manuals

The reference manual of commands has been named the *Commands Reference Manual* instead of the *User's Manual*. The section 8 manual pages (system maintenance commands and procedures) was moved back into the *Commands Reference Manual* because of popular demand. The major change to the reference manuals (*Commands Reference Manual* and *System Interface Manual*) is that the permuted index has been replaced by a new index of the familiar type found in ordinary books. The new index contains more locator material in many cases.

The list below is by no means extensive but contains the highlights of the changes made to reference manual pages since last release. Only substantive changes are listed here — formatting corrections and spelling corrections are not mentioned.

New Reference Manual Pages

Manual Page	*New or Changed*	*Description*

a.out.5	*changed*	Added description of how *dbx*(1) uses symbol table information.
adduser.8	*changed*	Completely rewrote manual page for clarity.
as.1	*changed*	Added description of **−J** and **−h** options.
cc.1	*changed*	Added descriptions of **−a** and **−v** options.
clocktool.1	*new*	Separated the description of the clocktool from the *suntools* manual page to make a standalone manual page.
crash.8s	*changed*	Replaced VAX list of traps with Sun list.
dbxtool.1	*new*	New window-based debug tool based on *dbx*.
des.1	*new*	encrypt or decrypt with Data Encryption Standard
dkinfo.8	*new*	Get information about disk geometry.
domainname.1	*new*	Get or set the domain name.
dump.8	*changed*	Clarifications of defaults.
ex.1	*changed*	Added more complete documentation on options.
fonttool.1	*new*	Font Editor
fstab.5	*changed*	Changed to reflect network file system capabilities.
ftpusers.5	*new*	Documented format of this file.
getdirentries.2	*new*	Get directory entries in an independent manner.
getdomainname.2	*new*	Get or set the name of a domain.
gettytab.5	*changed*	Added documentation of the 'td' (tab delay) capability.
gfxtool.1	*new*	Separated the description of the gfxtool from the *suntools* manual page to make a standalone manual page.
grep.1	*changed*	Changed cross-reference from *ex*(1) to *ed*(1).
icontool.1	*new*	Separated the description of the icontool from the *suntools* manual page to make a standalone manual page.
ie.4S	*new*	Description of Ethernet Interface.
indent.1	*changed*	Clarified caveats about in-place copying.
join.1	*changed*	Cleaned up bug in description of **−j** options.
ld.1	*changed*	New **−Ttext** and **−Tdata** options.
leave.1	*changed*	Added a bug report about relative time problems.
lookbib.1	*new*	Separated the *lookbib* and *indxbib* manual pages.
lpc.8	*changed*	Added documentation on the **topq** command; marked those *lpc* commands that only the super-user can use.
lpr.1	*changed*	Deleted reference to *cifplot* from list of options.
ls.1	*changed*	Added note about use of '=' sign to identify AF_UNIX domain sockets.
malloc.3	*changed*	Extensive revisions.
mem.4s	*changed*	Completely new manual page for VME bus memory.
mount.8	*changed*	Changed to reflect network file system capabilities.
netstat.8	*changed*	Clarified what the various options do.
onexit.3	*new*	Names functions to be called after calling *exit*(3).
perfmeter.1	*new*	Meter display of system performance values
perfmon.1	*new*	Graphical display of general system statistics
printcap.5	*changed*	Clarified use of **fs**, **fc**, **xs**, and **xx** options.
rastrepl.1	*new*	Replicate Raster Images.
rcp.1c	*changed*	Added bug note about third-party copying.
readnews.1	*changed*	Corrected error in description of **−a** option.
screendump.1	*changed*	Dump Screen to file in raster format.
screenload.1	*changed*	Load raster format file to screen.
sh.1	*changed*	Corrected example of setting variable from *pwd* command.

skyversion.8	*new*	Display the version number of the microcode in the SKY floating point board.
statfs.2	*new*	get file system statistics
stty.1	*changed*	Added information about setting speed to 19.2 KBaud.
sum.1	*changed*	Displays size of file in kilobytes not blocks.
tektool.1	*new*	Tektronix 4014 emulator tool.
tty.4	*changed*	Added information about setting speed to 19.2 KBaud.
vi.1	*changed*	Added more complete documentation on options.
vmstat.8	*changed*	Removed reference to 'simulating reference bits'.
vswap.1	*changed*	Changed syntax for pathnames.
w.1	*changed*	Clarified formats of time fields.
xt.4S	*new*	Description of 6250 bpi Tape Controller.
ypcat.1	*new*	print values in a YP data base
ypclnt.3n	*new*	yellow pages client interface
ypfiles.5	*new*	the yellowpages database and directory structure
ypinit.8	*new*	build and install yellow pages database
ypmake.8	*new*	rebuild yellow pages database
yppasswd.1	*new*	change login password in yellow pages
yppasswdd.8c	*new*	server for modifying yellow pages password file
yppush.8	*new*	yellow pages administration utilities
ypserv.8	*new*	yellow pages server and binder processes
ypwhich.8	*new*	which machines are the YP server and master?
zs.4s	*changed*	Now includes information on dial in and dial out support for the same line.

Part III — Differences Between Sun Release 2.0 and Berkeley 4.2 BSD

Although the Sun System evolved from Berkeley 4.2, there are differences between the two systems as they diverge from each other. Some of the differences are the inevitable result of differences between the VAX and the Sun hardware. Other differences arise as Sun people fix bugs, enhance utilities, and redo the documentation for a different market. This part of the document contains a description of differences between the Sun Release 2.0 and the Berkeley 4.2 release.

12. New, Changed, and Deleted Utilities

12.1. New Utilities

Sun System Release 2.0 contains the Network File System (NFS), the Yellow Pages (YP), network service, the Remote Procedure Call (RPC), and External Data Representation (XDR) facilities.

Utility packages and libraries include the **SunCore** graphics package — an implementation of the SIGGRAPH CORE graphics standard, the Computer Graphics Interface (CGI) graphics package, the SunWindows package for writing window-based tools in a multiple overlapping window environment. New tools this release are as described in part I of this document.

Differences in the C Compiler

Sun added the **-a** option to `cc` to enable test coverage.

Sun added the **-v** option to `cc` to get the compiler to print the command line of each separate phase as it is called.

Differences in dbx

Sun Microsystems has made extensive changes to the symbolic debugger **dbx**:

- ***dbx*** now displays structures, `char *` variables and function variables differently.
- includes a `debug` command to begin debugging a different program.
- includes an argument to the `where` command to print the top *n* elements of the stack trace.
- includes a `display` command to display certain variables when execution of the program being debugged stops.

- includes a `undisplay` command to undo the effect of the `display` command.
- includes a `dbxenv` command to control some of *dbx*'s actions. The `stringlen` option is the only one enabled so far. It controls how many characters of a `char *` variable are printed.
- Added an argument to the `next` and `step` commands to skip *n* lines.
- Added an `all` argument to the `delete` command to delete all breakpoints and traces.
- The `delete` command accepts a comma separated list of numbers as well as a single number.
- Enhanced the `help` command to give help for each individual command.

12.2. Deleted Utilities

User Contributed Software
portion of the 4.2 BSD release is not present in the Sun System.

`efl` — Extended FORTRAN Language.

`struct` — turn FORTRAN programs into Ratfor programs.

`fed` — font editor.

`quotas` The Sun System does not support disk quotas

`apply` There is no replacement for *apply* other than writing a shell script.

`sdb` — symbolic debugger — is replaced by `dbx`(1).

`style` `style` is now part of 'Writer's Workbench' which requires a supplementary license.

`diction` `diction` is now part of 'Writer's Workbench' which requires a supplementary license.

`explain` `explain` is now part of 'Writer's Workbench' which requires a supplementary license.

`learn` command is out of date and inappropriate for the Sun System.

`apropos` — can be achieved with a `man -k` command.

`cxref` — can be achieved with a `ctags -x` command.

`msgs` — use `news` instead.

`num` — can be achieved with a `cat -n` command.

`print` — can be achieved with a `lpr -p` command.

`see` — can be achieved with a `cat -v` command.

`tra` — can be achieved with a `tail -f` command.

finger was deleted from the Sun System because it is very specific to the campus environment at Berkeley.

13. Changes to Documentation

Documentation is changing the most. The massive changes in Sun's documentation are targeted towards a market consisting of people who are not UNIX experts, and who also can't look at the source code as the final arbiter of how something should work. Here are some of the details.

The Sun System documentation has gone through major changes — here is a brief summary of what's gone on and how this has affected the manuals. Basically, we've tried to improve the utility of the existing material by re-organizing and updating it. We've added new material to keep up with development, and to remedy the fact that many aspects of the existing system were never described. We'll begin with history and go to current organization and contents.

13.1. Original Organization

The original UNIX documentation was in four volumes known as The UNIX Programmer's Manual Volumes l, 2a, 2b, and 2c.

Volume 1 contained eight sections. Only sections 1, 6, and 7 were relevant to users — the rest was relevant to programmers writing code, or system managers looking after the system. Volume 1 was divided into sections like this: l: Commands — descriptions of the commands that a UNIX user uses on a daily basis; 2: System Calls — interfaces from running programs to the kernel; 3: Subroutines — access to the input-output system, the network library, the math library, the FORTRAN library, and miscellaneous libraries such as jobs and curses; 4: Device Descriptions — descriptions of the 'special files' that represent the interface from the device to the device driver; 5: File Formats — format of files such as ***a.out***, ***/etc/hosts***, and so on; 6: Games; 7: Tables — descriptions of manuscript-formatting macros (like **—ms**), maps of certain character codes (like the ASCII character set), and a diagram of the file system hierarchy; 8: Maintenance Commands and Procedures — the utilities that the super-user normally uses to keep the system running.

Volume 2 (a, b, and c) contained various tutorial-style papers on various UNIX utilities: the text-editor, the mail system, programming tools, and so on.

The above organization did not work very well — Volume 1 was too big to fit in a single three-inch binder; Volume 2 was a random collection of material that was often out of date or irrelevant.

13.2. New Organization of Manuals

Sun has 'unbundled' the previous documentation. Instead of four huge three-ring binders, there are now a number of smaller ones. These new manuals are focused on functional application areas — you should be able to find what you need without leafing through reams of irrelevant paper. There are index tabs between the sections.

13.2.1. Reference Manuals

The original Volume 1 was separated into three reference manuals, one for users, one for programmers, and one for system managers:

Commands Reference Manual
contains reference pages for all commands available to a user. This manual contains basic utility commands (the original section 1), games (the original section 6), descriptions of tables and macro packages (the original section 7), and maintenance commands and procedures for the system adminstrator (the original section 8).

System Interface Manual
covers areas of interest to programmers. It contains an overview of the system facilities, plus descriptions of system calls (the original section 2), descriptions of library routines (the original section 3), descriptions of the special files and devices (section 4), and descriptions of file formats (the original section 5). The old section 3 is divided by functional area.

System Manager's Manuals
There is a generic guide to installing the UNIX system, plus a large guide to system administration.

13.2.2. Supplementary Documentation

The original Volume 2 (2a, 2b, and 2c) has met with a similar fate. The manuals as they currently stand are:

Beginner's Guide to the Sun Workstation
A general overview and introduction to UNIX providing roadmaps and guides to the rest of the documentation, plus some tutorial introductions to the Sun System. It also contains a general guide to the communication facilities that UNIX offers you.

Editing and Text Processing on the Sun Workstation
documents the facilities that UNIX offers for massaging text files in various ways, and for processing documentation. This manual also introduces the basic document layout facilities.

Programming Tools for the Sun Workstation
contains information of general interest to anyone using UNIX to write programs. Contains UNIX Programming, Lint, Make, SCCS, DC, BC, M4 Macro Processor, LEX, YACC.

SunCore Programmer's Reference Manual
a reference description of Sun Microsystem's implementation of the ACM CORE graphics standard.

SunCGI Programmer's Reference Manual
a reference descrition of Sun Microsystem's implementation of the CGI (Computer Graphics Interface) — draft proposed standard.

SunWindows Programmer's Reference Manual
a reference description for programmers wishing to write tools to run in the Sun window system.

SunWindows Tutorial
a tutorial guide to writing tools for the Sun window system.

Networking on the Sun Workstation
Describes the Remote Procedure Call (RPC) and External Data Representation (XDR) facilities and also contains papers on Interprocess Communication and Network Implementation.

Device Drivers for the Sun Workstation
Contains much folklore on adding new device drivers to the system.

FORTRAN
Information specific to the FORTRAN programming language. Contains FORTRAN Programmer's Guide, Ratfor, and FORTRAN System Interface Routines.

Pascal
Information specific to the Pascal programming language. Contains Pascal User's Manual.