

**NAME**

sysconf – get configuration information at run time

**SYNOPSIS**

```
#include <unistd.h>
```

```
long sysconf(int name);
```

**DESCRIPTION**

POSIX allows an application to test at compile or run time whether certain options are supported, or what the value is of certain configurable constants or limits.

At compile time this is done by including `<unistd.h>` and/or `<limits.h>` and testing the value of certain macros.

At run time, one can ask for numerical values using the present function `sysconf()`. One can ask for numerical values that may depend on the filesystem in which a file resides using `fpathconf(3)` and `pathconf(3)`. One can ask for string values using `confstr(3)`.

The values obtained from these functions are system configuration constants. They do not change during the lifetime of a process.

For options, typically, there is a constant `_POSIX_FOO` that may be defined in `<unistd.h>`. If it is undefined, one should ask at run time. If it is defined to `-1`, then the option is not supported. If it is defined to `0`, then relevant functions and headers exist, but one has to ask at run time what degree of support is available. If it is defined to a value other than `-1` or `0`, then the option is supported. Usually the value (such as `200112L`) indicates the year and month of the POSIX revision describing the option. Glibc uses the value `1` to indicate support as long as the POSIX revision has not been published yet. The `sysconf()` argument will be `_SC_FOO`. For a list of options, see `posixoptions(7)`.

For variables or limits, typically, there is a constant `_FOO`, maybe defined in `<limits.h>`, or `_POSIX_FOO`, maybe defined in `<unistd.h>`. The constant will not be defined if the limit is unspecified. If the constant is defined, it gives a guaranteed value, and a greater value might actually be supported. If an application wants to take advantage of values which may change between systems, a call to `sysconf()` can be made. The `sysconf()` argument will be `_SC_FOO`.

**POSIX.1 variables**

We give the name of the variable, the name of the `sysconf()` argument used to inquire about its value, and a short description.

First, the POSIX.1 compatible values.

**ARG\_MAX - \_SC\_ARG\_MAX**

The maximum length of the arguments to the `exec(3)` family of functions. Must not be less than `_POSIX_ARG_MAX` (4096).

**CHILD\_MAX - \_SC\_CHILD\_MAX**

The maximum number of simultaneous processes per user ID. Must not be less than `_POSIX_CHILD_MAX` (25).

**HOST\_NAME\_MAX - \_SC\_HOST\_NAME\_MAX**

Maximum length of a hostname, not including the terminating null byte, as returned by `gethostname(2)`. Must not be less than `_POSIX_HOST_NAME_MAX` (255).

**LOGIN\_NAME\_MAX - \_SC\_LOGIN\_NAME\_MAX**

Maximum length of a login name, including the terminating null byte. Must not be less than `_POSIX_LOGIN_NAME_MAX` (9).

**NGROUPS\_MAX - \_SC\_NGROUPS\_MAX**

Maximum number of supplementary group IDs.

**clock ticks - \_SC\_CLK\_TCK**

The number of clock ticks per second. The corresponding variable is obsolete. It was of course called `CLK_TCK`. (Note: the macro `CLOCKS_PER_SEC` does not give information: it must

equal 1000000.)

**OPEN\_MAX - \_SC\_OPEN\_MAX**

The maximum number of files that a process can have open at any time. Must not be less than **\_POSIX\_OPEN\_MAX** (20).

**PAGESIZE - \_SC\_PAGESIZE**

Size of a page in bytes. Must not be less than 1.

**PAGE\_SIZE - \_SC\_PAGE\_SIZE**

A synonym for **PAGESIZE/\_SC\_PAGESIZE**. (Both **PAGESIZE** and **PAGE\_SIZE** are specified in POSIX.)

**RE\_DUP\_MAX - \_SC\_RE\_DUP\_MAX**

The number of repeated occurrences of a BRE permitted by **regex(3)** and **regcomp(3)**. Must not be less than **\_POSIX2\_RE\_DUP\_MAX** (255).

**STREAM\_MAX - \_SC\_STREAM\_MAX**

The maximum number of streams that a process can have open at any time. If defined, it has the same value as the standard C macro **FOPEN\_MAX**. Must not be less than **\_POSIX\_STREAM\_MAX** (8).

**SYMLOOP\_MAX - \_SC\_SYMLOOP\_MAX**

The maximum number of symbolic links seen in a pathname before resolution returns **ELOOP**. Must not be less than **\_POSIX\_SYMLOOP\_MAX** (8).

**TTY\_NAME\_MAX - \_SC\_TTY\_NAME\_MAX**

The maximum length of terminal device name, including the terminating null byte. Must not be less than **\_POSIX\_TTY\_NAME\_MAX** (9).

**TZNAME\_MAX - \_SC\_TZNAME\_MAX**

The maximum number of bytes in a timezone name. Must not be less than **\_POSIX\_TZNAME\_MAX** (6).

**\_POSIX\_VERSION - \_SC\_VERSION**

indicates the year and month the POSIX.1 standard was approved in the format **YYMM**; the value **199009L** indicates the Sept. 1990 revision.

**POSIX.2 variables**

Next, the POSIX.2 values, giving limits for utilities.

**BC\_BASE\_MAX - \_SC\_BC\_BASE\_MAX**

indicates the maximum *obase* value accepted by the **bc(1)** utility.

**BC\_DIM\_MAX - \_SC\_BC\_DIM\_MAX**

indicates the maximum value of elements permitted in an array by **bc(1)**.

**BC\_SCALE\_MAX - \_SC\_BC\_SCALE\_MAX**

indicates the maximum *scale* value allowed by **bc(1)**.

**BC\_STRING\_MAX - \_SC\_BC\_STRING\_MAX**

indicates the maximum length of a string accepted by **bc(1)**.

**COLL\_WEIGHTS\_MAX - \_SC\_COLL\_WEIGHTS\_MAX**

indicates the maximum numbers of weights that can be assigned to an entry of the **LC\_COLLATE order** keyword in the locale definition file,

**EXPR\_NEST\_MAX - \_SC\_EXPR\_NEST\_MAX**

is the maximum number of expressions which can be nested within parentheses by **expr(1)**.

**LINE\_MAX - \_SC\_LINE\_MAX**

The maximum length of a utility's input line, either from standard input or from a file. This includes space for a trailing newline.

**RE\_DUP\_MAX - \_SC\_RE\_DUP\_MAX**

The maximum number of repeated occurrences of a regular expression when the interval notation  $\{m,n\}$  is used.

**POSIX2\_VERSION - \_SC\_2\_VERSION**

indicates the version of the POSIX.2 standard in the format of YYYYMMML.

**POSIX2\_C\_DEV - \_SC\_2\_C\_DEV**

indicates whether the POSIX.2 C language development facilities are supported.

**POSIX2\_FORT\_DEV - \_SC\_2\_FORT\_DEV**

indicates whether the POSIX.2 FORTRAN development utilities are supported.

**POSIX2\_FORT\_RUN - \_SC\_2\_FORT\_RUN**

indicates whether the POSIX.2 FORTRAN run-time utilities are supported.

**\_POSIX2\_LOCALEDEF - \_SC\_2\_LOCALEDEF**

indicates whether the POSIX.2 creation of locales via **localedef**(1) is supported.

**POSIX2\_SW\_DEV - \_SC\_2\_SW\_DEV**

indicates whether the POSIX.2 software development utilities option is supported.

These values also exist, but may not be standard.

**- \_SC\_PHYS\_PAGES**

The number of pages of physical memory. Note that it is possible for the product of this value and the value of **\_SC\_PAGESIZE** to overflow.

**- \_SC\_AVPHYS\_PAGES**

The number of currently available pages of physical memory.

**- \_SC\_NPROCESSORS\_CONF**

The number of processors configured. See also **get\_nprocs\_conf**(3).

**- \_SC\_NPROCESSORS\_ONLN**

The number of processors currently online (available). See also **get\_nprocs\_conf**(3).

**RETURN VALUE**

The return value of **sysconf**() is one of the following:

- \* On error,  $-1$  is returned and *errno* is set to indicate the cause of the error (for example, **EINVAL**, indicating that *name* is invalid).
- \* If *name* corresponds to a maximum or minimum limit, and that limit is indeterminate,  $-1$  is returned and *errno* is not changed. (To distinguish an indeterminate limit from an error, set *errno* to zero before the call, and then check whether *errno* is nonzero when  $-1$  is returned.)
- \* If *name* corresponds to an option, a positive value is returned if the option is supported, and  $-1$  is returned if the option is not supported.
- \* Otherwise, the current value of the option or limit is returned. This value will not be more restrictive than the corresponding value that was described to the application in *<unistd.h>* or *<limits.h>* when the application was compiled.

**ERRORS****EINVAL**

*name* is invalid.

**ATTRIBUTES**

For an explanation of the terms used in this section, see **attributes**(7).

Interface	Attribute	Value
<b>sysconf</b> ()	Thread safety	MT-Safe env

**CONFORMING TO**

POSIX.1-2001, POSIX.1-2008.

**BUGS**

It is difficult to use **ARG\_MAX** because it is not specified how much of the argument space for **exec(3)** is consumed by the user's environment variables.

Some returned values may be huge; they are not suitable for allocating memory.

**SEE ALSO**

**bc(1)**, **expr(1)**, **getconf(1)**, **locale(1)**, **confstr(3)**, **fpathconf(3)**, **pathconf(3)**, **posixoptions(7)**

**COLOPHON**

This page is part of release 5.05 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at <https://www.kernel.org/doc/man-pages/>.