

**NAME**

**getrusage** - get information about resource utilization

**LIBRARY**

Standard C Library (libc, -lc)

**SYNOPSIS**

```
#include <sys/types.h>
```

```
#include <sys/time.h>
```

```
#include <sys/resource.h>
```

```
#define RUSAGE_SELF 0
```

```
#define RUSAGE_CHILDREN -1
```

```
#define RUSAGE_THREAD 1
```

*int*

```
getrusage(int who, struct rusage *rusage);
```

**DESCRIPTION**

The **getrusage()** system call returns information describing the resources utilized by the current thread, the current process, or all its terminated child processes. The *who* argument is either **RUSAGE\_THREAD**, **RUSAGE\_SELF**, or **RUSAGE\_CHILDREN**. The buffer to which *rusage* points will be filled in with the following structure:

```
struct rusage {
    struct timeval ru_utime; /* user time used */
    struct timeval ru_stime; /* system time used */
    long ru_maxrss;        /* max resident set size */
    long ru_ixrss;         /* integral shared text memory size */
    long ru_idrss;         /* integral unshared data size */
    long ru_isrss;         /* integral unshared stack size */
    long ru_minflt;        /* page reclaims */
    long ru_majflt;        /* page faults */
    long ru_nswap;         /* swaps */
    long ru_inblock;       /* block input operations */
    long ru_oublock;       /* block output operations */
    long ru_msgsnd;        /* messages sent */
    long ru_msgrcv;        /* messages received */
    long ru_nsignals;      /* signals received */
    long ru_nvcsw;         /* voluntary context switches */
};
```

```
    long ru_nivcsw;    /* involuntary context switches */  
};
```

The fields are interpreted as follows:

- ru\_utime* the total amount of time spent executing in user mode.
- ru\_stime* the total amount of time spent in the system executing on behalf of the process(es).
- ru\_maxrss* the maximum resident set size utilized (in kilobytes).
- ru\_ixrss* an "integral" value indicating the amount of memory used by the text segment that was also shared among other processes. This value is expressed in units of kilobytes \* ticks-of-execution. Ticks are statistics clock ticks. The statistics clock has a frequency of **sysconf(\_SC\_CLK\_TCK)** ticks per second.
- ru\_idrss* an integral value of the amount of unshared memory residing in the data segment of a process (expressed in units of kilobytes \* ticks-of-execution).
- ru\_isrss* an integral value of the amount of unshared memory residing in the stack segment of a process (expressed in units of kilobytes \* ticks-of-execution).
- ru\_minflt* the number of page faults serviced without any I/O activity; here I/O activity is avoided by "reclaiming" a page frame from the list of pages awaiting reallocation.
- ru\_majflt* the number of page faults serviced that required I/O activity.
- ru\_nswap* the number of times a process was "swapped" out of main memory.
- ru\_inblock* the number of times the file system had to perform input.
- ru\_oublock* the number of times the file system had to perform output.
- ru\_msgsnd* the number of IPC messages sent.
- ru\_msgrcv* the number of IPC messages received.
- ru\_nsignals* the number of signals delivered.
- ru\_nvcsw* the number of times a context switch resulted due to a process voluntarily giving up the

processor before its time slice was completed (usually to await availability of a resource).

*ru\_nivcsw* the number of times a context switch resulted due to a higher priority process becoming runnable or because the current process exceeded its time slice.

## NOTES

The numbers *ru\_inblock* and *ru\_oublock* account only for real I/O; data supplied by the caching mechanism is charged only to the first process to read or write the data.

## RETURN VALUES

The **getrusage()** function returns the value 0 if successful; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

## ERRORS

The **getrusage()** system call will fail if:

[EINVAL] The *who* argument is not a valid value.

[EFAULT] The address specified by the *rusage* argument is not in a valid part of the process address space.

## SEE ALSO

gettimeofday(2), wait(2), clocks(7)

## HISTORY

The **getrusage()** system call appeared in 4.2BSD. The RUSAGE\_THREAD facility first appeared in FreeBSD 8.1.

## BUGS

There is no way to obtain information about a child process that has not yet terminated.