NAME

wait − wait for process to die

SYNOPSIS

wait (&status)
struct { char lobyte; char hibyte; } status;

DESCRIPTION

Wait causes its caller to delay until one of its child processes terminates. If any child has died since the last *wait*, return is immediate; if there are no children, return is immediate with the error bit set (resp. with a value of −1 returned). In the case of several children several *wait* calls are needed to learn of all the deaths.

If no error is indicated on return, the r1 high byte, i.e. *status.hibyte*, contains the low byte of the child process r0, i.e. the argument of *exit*, when it terminated. The r1 low byte, i.e. *status.lobyte*, contains the termination status of the process. See *signal*(2) for a list of termination statuses (signals); **0** status indicates normal termination. If the **0200** bit of the termination status is set, a core image of the process was produced by the system. Status **0177** is returned for a stopped process which has not terminated and can be restarted (see *ptrace*(2)).

On return, r0 contains the process ID of the dead child. From C, the process ID of the child is the returned value.

SEE ALSO

exit(2), fork(2), signal(2)

DIAGNOSTICS

The error bit (c-bit) on if no children not previously waited for. From C, a returned value of −1 indicates an error.

ASSEMBLER

(wait = 7.)
sys wait
(process id in r0)
(status in r1)