NAME break -- set program break

SYNOPSIS sys break; addr / break = 17.

DESCRIPTION _break_ sets the system's idea of the highest location used by the program to _addr_. Locations greater than _addr_ and below the stack pointer are not swapped and are thus liable to unexpected modification.

If the argument is 0 or higher than the stack pointer the entire 4K word user core area is swapped.

When a program begins execution via _exec_ the break is set, at the highest location defined by the program and data storage areas. Ordinarily, therefore, only programs with growing data areas need to use break.

FILES

SEE ALSO exec

DIAGNOSTICS none; strange addresses cause the break to be set to include all of core.

BUGS

OWNER ken, dmr

NAME cemt -- catch emt traps

SYNOPSIS sys cemt; arg / cemt = 29.; not in assembler

DESCRIPTION This call allows one to catch traps resulting from the emt instruction. Arg is a location within the program; emt traps are sent to that location. The normal effect of emt traps may be restored by giving an arg equal to 0.

Prior to the use of this call, the result of an emt instruction is a simulated rts instruction. The operand field is interpreted as a register, and an rts instruction is simulated for that register (after verifying that various registers have appropriate values). This feature is useful for debugging, since the most dangerous program bugs usually involve an rts with bad data on the stack or in a register.

FILES

SEE ALSO

DIAGNOSTICS

BUGS

OWNER ken, dmr

NAME chdir -- change working directory

SYNOPSIS sys chdir; dirname / chdir = 12.

DESCRIPTION dirname is address of the pathname of a directory, terminated by a 0 byte. chdir causes this directory to become the current working directory.

FILES

SEE ALSO

DIAGNOSTICS The error bit (c—bit) is set if the given name is not that of a directory.

BUGS

OWNER ken, dmr

NAME chmod -- change mode of file

SYNOPSIS

```
sys     chmod; name; mode          / chmod = 15.
```

DESCRIPTION The file whose name is given as the null-terminated string pointed to by name has its mode changed to _mode_. Modes are constructed by _oring_ together some combination of the following:

```
01 write, non-owner
02 read, non-owner
04 write, owner
10 read, owner
20 executable
40 set user ID on execution
```

Only the owner of a file (or the super-user) may change the mode.

FILES

SEE ALSO

DIAGNOSTICS Error bit (c-bit) set if _name_ cannot be found or if current user is neither the owner of the file nor the super-user.

BUGS

OWNER ken, dmr

NAME chown -- change owner of file

SYNOPSIS

```
sys     chown; name; owner          / chown = 16.
```

DESCRIPTION The file whose name is given by the null–terminated string pointed to by name has its owner changed to owner. Only the present owner of a file (or the super–user) may donate the file to another user. Also, one may not change the owner of a file with the set–user–ID bit on, otherwise one could create Trojan Horses able to misuse other's files.

FILES

SEE ALSO /etc/uids has the mapping between user names and user numbers.

DIAGNOSTICS The error bit (c–bit) is set on illegal owner changes.

BUGS

OWNER ken, dmr

NAME close -- close a file

SYNOPSIS (file descriptor in r0)

```
sys     close          / close = 6.
```

DESCRIPTION Given a file descriptor such as returned from an open or creat call, close closes the associated file. A close of all files is automatic on exit, but since processes are limited to 10 simultaneously open files, close is necessary to programs which deal with many files.

FILES

SEE ALSO creat, open

DIAGNOSTICS The error bit (c—bit) is set for an unknown file descriptor.

BUGS

OWNER ken, dmr

NAME creat -- create a new file

SYNOPSIS

```
sys     creat; name; mode          / creat = 8.
(file descriptor in r0)
```

DESCRIPTION creat creates a new file or prepares to rewrite an existing file called name; name is the address of a null-terminated string. If the file did not exist, it is given mode mode; if it did exist, its mode and owner remain unchanged but it is truncated to 0 length.

The file is also opened for writing, and its file descriptor is returned in r0.

The mode given is arbitrary; it need not allow writing. This feature is used by programs which deal with temporary files of fixed names. The creation is done with a mode that forbids writing. Then if a second instance of the program attempts a creat, an error is returned and the program knows that the name is unusable for the moment. If the last link to an open file is removed, the file is not destroyed until the file is closed.

FILES

SEE ALSO write, close

DIAGNOSTICS The error bit (c-bit) may be set if: a needed directory is not readable; the file does not exist and the directory in which it is to be created is not writable; the file does exist and is unwritable; the file is a directory.

B UGS

OWNER ken, dmr

NAME exec --execute a file

SYNOPSIS

```
sys    exec; name; args    / exec = 11.

        name: <...\0>
        ...
args: arg1; arg2; ...; 0
arg1: <...\0>
...
```

DESCRIPTION

exec overlays the calling process with the named file, then transfers to the beginning of the core image of the file. The first argument to <u>exec</u> is a pointer to the name of the file to be executed. The second is the address of a list of pointers to arguments to be passed to the file. Conventionally, the first argument is the name of the file. Each pointer addresses a string terminated by a null byte.

There can be no return from the file; the calling core image is lost.

The program break is set from the executed file; see the format of a.out.

Once the called file starts execution, the arguments are passed as follows. The stack pointer points to the number of arguments. Just above this number is a list of pointers to the argument strings.

```
  sp—> nargs

          arg1
          ...
          argn

  arg1: <arg1\0>
     ...
argn: <argn\0>
```

The arguments are placed as high as possible incore: just below 60000(8).

Files remain open across exec calls. However, the illegal instruction, emt, quit, and interrupt trap specifications are reset to the standard values. (See <u>ilgins</u>, cemt, intr).

Each user has a <u>real</u> user ID and an <u>effective</u> (The real ID identifies the person using the system; the effective ID determines his access privileges.) exec changes the effective user ID to the owner of the executed file if the file has the "set—user—ID mode. The real user ID is not affected.

FILES

SEE ALSO fork

DIAGNOSTICS If the file cannot be read or if it is not executable, a return from *exec* constitutes the diagnostic. The error bit (c—bit) is set.

BUGS

OWNER ken, dmr

NAME exit -- terminate process

SYNOPSIS sys exit / exit = I

DESCRIPTION _exit_ is the normal means of terminating a process. All files are closed and the parent process is notified if it is executing a wait.

This call can never return.

FILES

SEE ALSO sys wait

DIAGNOSTICS --

BUGS

OWNER ken, dmr

NAME fork -- spawn new process

SYNOPSIS

```
sys     fork     / fork = 2.
(new process return)
(old process return)
```

DESCRIPTION _fork_ is the only way new processes are created. The new process's core image is a copy of that of the caller of _fork_ the only distinction is the return location and the fact that r0 in the old process contains the process ID of the new process. This process ID is used by _wait_.

FILES

SEE ALSO sys wait, sys exec

DIAGNOSTICS The error bit (c—bit) is set in the old process if a new process could not be created because of lack of swap space.

BUGS See _wait_ for a subtle bug in process destruction.

OWNER ken, dmr

NAME fstat -- get status of open file

SYNOPSIS (file descriptor in r0)

```
sys     fstat; buf        / fstat = 28.
```

DESCRIPTION This call is identical to stat, except that it operates on open files instead of files given by name. It is most often used to get the status of the standard input and output files, whose names are unknown.

FILES

SEE ALSO sys stat

DIAGNOSTICS The error bit (c—bit) is set if the file descriptor is unknown.

BUGS

OWNER ken, dmr

NAME getuid -- get user identification

SYNOPSIS

```
sys      getuid / getuid = 24.
(user ID in r0)
```

DESCRIPTION _getuid_ returns `the real user ID of the current process. The real user ID identifies the person who is logged in, in contradistinction to the effective user ID, which determines his access permission at each moment. It is thus useful to programs which operate using the "set user ID" mode, to find out who invoked them.

FILES /etc/uids can be used to map the user ID number into a name.

SEE ALSO setuid

DIAGNOSTICS

BUGS

OWNER ken, dmr

NAME gtty -- get typewriter status

SYNOPSIS

```
(file descriptor in r0)
sys gtty; arg / gtty = 32.; not in assembler
...
arg: .=.+6
```

DESCRIPTION *gtty* stores in the three words addressed by arg the status of the typewriter whose file descriptor is given in r0. The format is the same as that passed by *stty*.

FILES

SEE ALSO stty

DIAGNOSTICS Error bit (c—bit) is set if the file descriptor does not refer to a typewriter.

BUGS

OWNER ken, dmr

NAME ilgins -- catch illegal instruction trap

SYNOPSIS sys ilgins; arg / ilgins = 33.; not in assembler

DESCRIPTION ilgins allows a program to catch illegal instruction traps. If arg is zero, the normal instruction trap handling is done: the process is terminated and a core image is produced. If arg is a location within the program, control is passed to arg when the trap occurs.

This call is used to implement the floating point simulator, which catches and interprets 11/45 floating point instructions.

FILES

SEE ALSO fptrap, the floating point package

DIAGNOSTICS

BUGS

OWNER ken, dmr

NAME intr -- set interrupt handling

SYNOPSIS sys intr; arg / intr = 27.

DESCRIPTION When *arg* is 0, interrupts (ASCII DELETE) are ignored. When *arg* is 1, interrupts cause their normal result, that is, force an exit. When *arg* is a location within the program, control is transferred to that location when an interrupt occurs.

After an interrupt is caught, it is possible to resume execution by means of an *rti* instruction; however, great care must be exercised, since all I/O is terminated abruptly upon an interrupt. In particular, reads of the typewriter tend to return with 0 characters read, thus simulating an end of file.

FILES

SEE ALSO quit

DIAGNOSTICS

BUGS It should be easier to resume after an interrupt, but I don't know how to make it work.

OWNER ken, dmr

NAME link -- link to a file

SYNOPSIS sys link; name1 name2 / link = 9.

DESCRIPTION A link to name1 is created; the link has name name2. Either name may be an arbitrary path name.

FILES

SEE ALSO unlink

DIAGNOSTICS The error bit (c—bit) is set when name cannot be found; when name2 already exists; when the directory of name1 cannot be written; when an attempt is made to link to a directory by a user other than the super—user.

BUGS

OWNER ken, dmr

NAME mkdir -- make a directory

SYNOPSIS sys mkdir; name; mode / mkdir = 14.

DESCRIPTION _mkdir_ creates an empty directory whose name is the null–terminated string pointed to by name. The mode of the di rectory is mode. The special entries "." and ".." are not present.

mkdir can only be invoked by the super–user.

FILES

SEE ALSO mkdir command

DIAGNOSTICS Error bit (c–bit) is set if the directory already exists or if the user is not the super–user.

B UGS

OWNER ken, dmr