任务介绍

你可以在本机上检查自己的这几个文件和目录的格式，然后在 Starry 中实现时查询 axmem / axprocess 等等模块，填入同样的信息。不需要完全支持，

- `/proc/filesystems` 就写 fat32，或者考虑到以后我们有 ext4 也可以再加一个
- `/proc/self/status` 支持 name / pid / fdsize 应该足够
- `/sys/devices/system/cpu` 只会被访问目录项和之前就支持的 `/sys/devices/system/cpu/online` 文件。ffmpeg 只是"看看"这个目录，不会实际打开其他文件读写。所以只需要在里面放 `cpu0`，`cpu1`... 这些空目录，再加上上述的 `online` 文件就行。

相关截图

```
[ 78.384916 0:8 linux_syscall_api::syscall:31] [syscall] id = OPENAT, args = [4294967196, 184493943, 524288, 0, 8, 1], entry
[ 78.387707 0:8 linux_syscall_api::syscall_fs::imp::io:410] path: "/proc/filesystems"
[ 78.389937 0:8 axprocess::link:243] create_link: /proc/filesystems → /proc/filesystems
[ 78.391939 0:8 linux_syscall_api::syscall:53] [syscall] id = 257,return 4
[ 78.393988 0:8 axtrap::arch::x86_64:28] User #PF @ 0x1ff8fd5, fault_vaddr=0x212942a, error_code=0x4
[ 78.396913 0:8 axtrap::arch::x86_64:28] User #PF @ 0x1fd9250, fault_vaddr=0x1fd9250, error_code=0x14
[ 78.399617 0:8 axtrap::arch::x86_64:28] User #PF @ 0x1fd00b0, fault_vaddr=0x1fd00b0, error_code=0x14
[ 78.402756 0:8 axtrap::arch::x86_64:28] User #PF @ 0x1fceb10, fault_vaddr=0x1fceb10, error_code=0x14
[ 78.405520 0:8 axtrap::arch::x86_64:28] User #PF @ 0x1fdaeb0, fault_vaddr=0x1fdaeb0, error_code=0x14
[ 78.408134 0:8 axtrap::arch::x86_64:28] User #PF @ 0x2063cd0, fault_vaddr=0x2063cd0, error_code=0x14
[ 78.410852 0:8 linux_syscall_api::syscall:31] [syscall] id = FSTATAT, args = [4, 34767951, 1073740240, 4096, 0, 1], entry
[ 78.414452 0:8 linux_syscall_api::syscall_fs::imp::stat:78] path : /proc/filesystems
[ 78.416396 0:8 linux_syscall_api::syscall_fs::ctype::mount:106] get_stat_in_fs: /proc/filesystems
[ 78.418763 0:8 linux_syscall_api::syscall:53] [syscall] id = 262,return 0
[ 78.420638 0:8 axtrap::arch::x86_64:28] User #PF @ 0x1ff48b0, fault_vaddr=0xcfdaf28, error_code=0x6
[ 78.424016 0:8 linux_syscall_api::syscall:31] [syscall] id = READ, args = [4, 217943840, 8192, 35040544, 4, 217943840], entry
[ 78.428128 0:8 linux_syscall_api::syscall_fs::imp::io:31] [read()] fd: 4, buf: 0xcfd8f20, len: 8192
```

```
[ 79.540857 0:8 linux_syscall_api::syscall:31] [syscall] id = OPENAT, args = [4294967196, 180077338, 0, 0, 8, 1], entry
[ 79.544517 0:8 linux_syscall_api::syscall_fs::imp::io:410] path: "/proc/self/status"
[ 79.546494 0:8 axprocess::link:243] create_link: /proc/self/status → /proc/self/status
[ 79.548728 0:8 linux_syscall_api::syscall:53] [syscall] id = 257,return 4
[ 79.550806 0:8 linux_syscall_api::syscall:31] [syscall] id = FSTATAT, args = [4, 34767951, 1073740320, 4096, 0, 1], entry
[ 79.553894 0:8 linux_syscall_api::syscall_fs::imp::stat:78] path : /proc/self/status
[ 79.555865 0:8 linux_syscall_api::syscall_fs::ctype::mount:106] get_stat_in_fs: /proc/self/status
[ 79.558104 0:8 linux_syscall_api::syscall:53] [syscall] id = 262,return 0
[ 79.560364 0:8 linux_syscall_api::syscall:31] [syscall] id = READ, args = [4, 217944784, 8192, 119, 4, 217944784], entry
[ 79.563209 0:8 linux_syscall_api::syscall_fs::imp::io:31] [read()] fd: 4, buf: 0xcfd92d0, len: 8192
[ 79.565664 0:8 linux_syscall_api::syscall:53] [syscall] id = 0,return 0
[ 79.567545 0:8 linux_syscall_api::syscall:31] [syscall] id = CLOSE, args = [4, 4222428184, 35023360, 119, 4, 217944784], entry
```

```
[ 89.774918 0:8 linux_syscall_api::syscall:31] [syscall] id = OPENAT, args = [4294967196, 34799040, 524288, 0, 524288, 34799040], entry
[ 89.778420 0:8 linux_syscall_api::syscall_fs::imp::io:410] path: "/sys/devices/system/cpu/online"
[ 89.780779 0:8 axprocess::link:243] create_link: /sys/devices/system/cpu/online → /sys/devices/system/cpu/online
[ 89.783349 0:8 linux_syscall_api::syscall:53] [syscall] id = 257,return 5
[ 89.785488 0:8 linux_syscall_api::syscall:31] [syscall] id = READ, args = [5, 1073736848, 1024, 0, 524288, 34799040], entry
[ 89.788855 0:8 linux_syscall_api::syscall_fs::imp::io:31] [read()] fd: 5, buf: 0x3fffec90, len: 1024
[ 89.791666 0:8 linux_syscall_api::syscall:53] [syscall] id = 0,return 3
[ 89.793550 0:8 linux_syscall_api::syscall:31] [syscall] id = CLOSE, args = [5, 10, 1073736851, 34663104, 0, 0], entry
[ 89.796478 0:8 linux_syscall_api::syscall_fs::imp::io:460] Into syscall_close. fd: 5
[ 89.799037 0:8 linux_syscall_api::syscall:53] [syscall] id = 3,return 0
```

修改代码

crates/axfs/src/mounts.rs

```
use alloc::sync::Arc;
use axfs_vfs::{VfsNodeType, VfsOps, VfsResult};
use axtask::current;

use crate::alloc::string::ToString;
use crate::fs;
```

```
// Create /proc/self/stat
proc_root.create("self", VfsNodeType::Dir)?;
proc_root.create("self/stat", VfsNodeType::File)?;
proc_root.create("self/exe", VfsNodeType::File)?;
proc_root.create("self/status", VfsNodeType::File)?;
let current_task = current();
let name = current_task.name().as_bytes();
```

```
let getid = current_task.id().as_u64().to_string();
let id = getid.as_bytes();
let file_status = proc_root.clone().lookup("./self/status")?;
file_status.write_at(0, name)?;
file_status.write_at(name.len().try_into().unwrap(), b"\n")?;
file_status.write_at((name.len() + 1).try_into().unwrap(), id)?;
file_status.write_at((name.len() + id.len() +1).try_into().unwrap(),
b"\n256\n")?;

// Create /proc/filesystems
proc_root.create("filesystems", VfsNodeType::File)?;
let file_fs = proc_root.clone().lookup("./filesystems")?;
file_fs.write_at(0, b"fat32\next4\n")?;
```

执行效果

```
/ # cd /proc/self/
/proc/self/ # busybox cat status
main
3
256
```

当前问题

需要添加axfs的依赖axtask，在multitask的情况下能够正常使用current()，但是signal-task时，axtask/api不可用，无法获取当前task的信息。需要考虑starry的设计，之后开启支持zlm工作时要进行讨论。