

# **Implementace PROC FS**

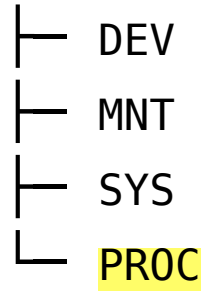
Vladan Trhlík

# Cíl

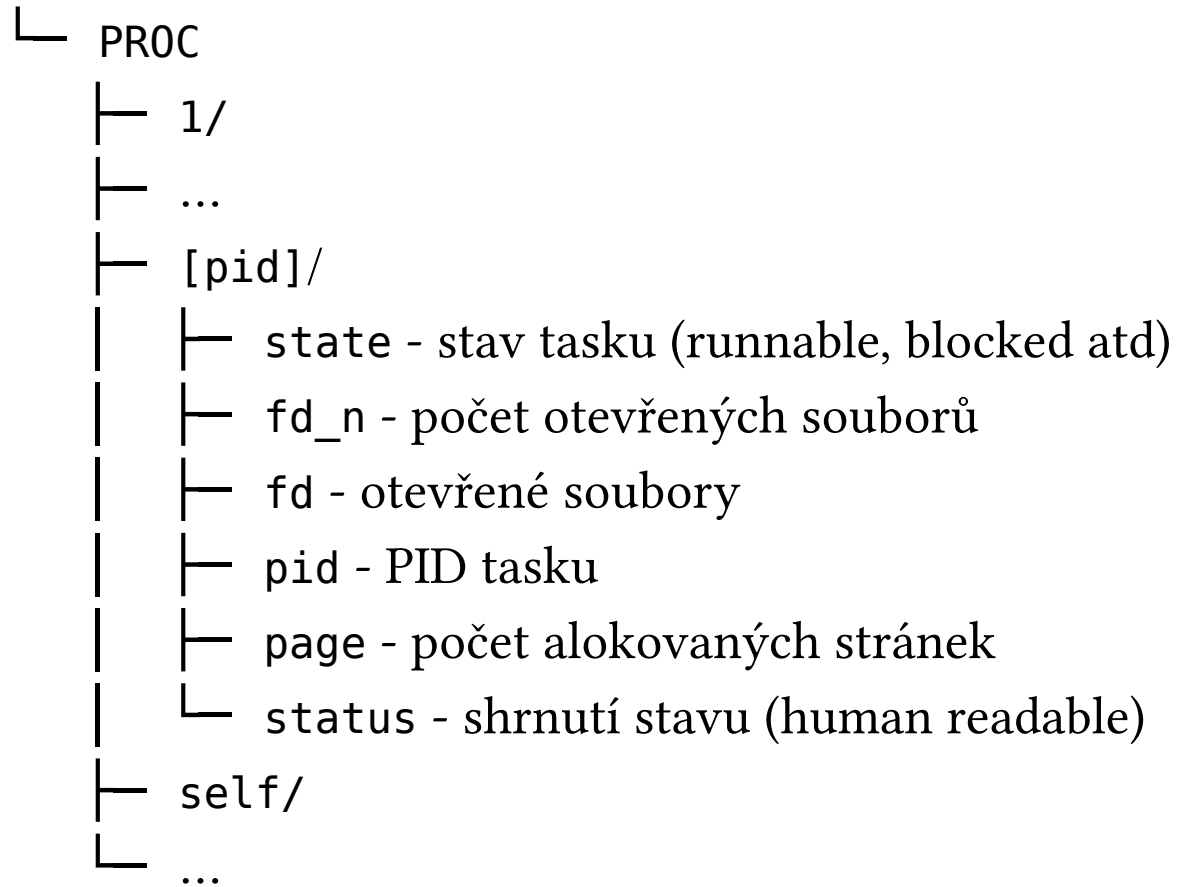
- vytvoření procfs mount point v dosavadním FS KIV-RTOS
- musí obsahovat:
  - aktuálních počet tasků
  - informace plánovače
  - celkový počet otevřených souborů
  - informace ke každému tasku (PID, počet otevřených souborů...)

# **Struktura FS**

# Přidání mount pointu



# Složky procesů



# Informace o systému

- └ PROC
  - └ ...
  - └ sched - počet runnable, blocked tasků
  - └ tasks - aktuální počet tasků
  - └ ticks - počet tiků od startu (obodoba /proc/uptime)

**Implementace**

## Mount point

- fs/filesystem.h – CFileSystem:

```
143  TFS_Tree_Node mRoot_Dev;  
144  TFS_Tree_Node mRoot_Sys;  
145  TFS_Tree_Node mRoot_Mnt;  
146  TFS_Tree_Node mRoot_Proc;
```



## FS Driver

- fs/drivers/proc\_fs.h – třída driveru

```
class CProc_FS_Driver : public IFilesystem_Driver {  
    IFile* Open_File(const char* path, NFile_Open_Mode mode)  
    { ... }  
}
```

## Soubor Tasku

```
class CProcFS_PID_File final : public IFile {  
    CProcFS_PID_File(int pid, NProcFS_PID_Type type) { ... }  
    uint32_t Read(char* buffer, uint32_t num) { ... }  
}  
  
enum NProcFS_PID_Type {  
    PID, STATE, FD_N, FD, STATUS, PAGE  
};
```

## Soubor systémové informace

```
class CProcFS_Status_File final : public IFile
{
    CProcFS_Status_File(NProcFS_Status_Type type) { ... }
    uint32_t Read(char* buffer, uint32_t num) { ... }
}

enum NProcFS_Status_Type {
    SCHED, TASKS, TICKS
};
```