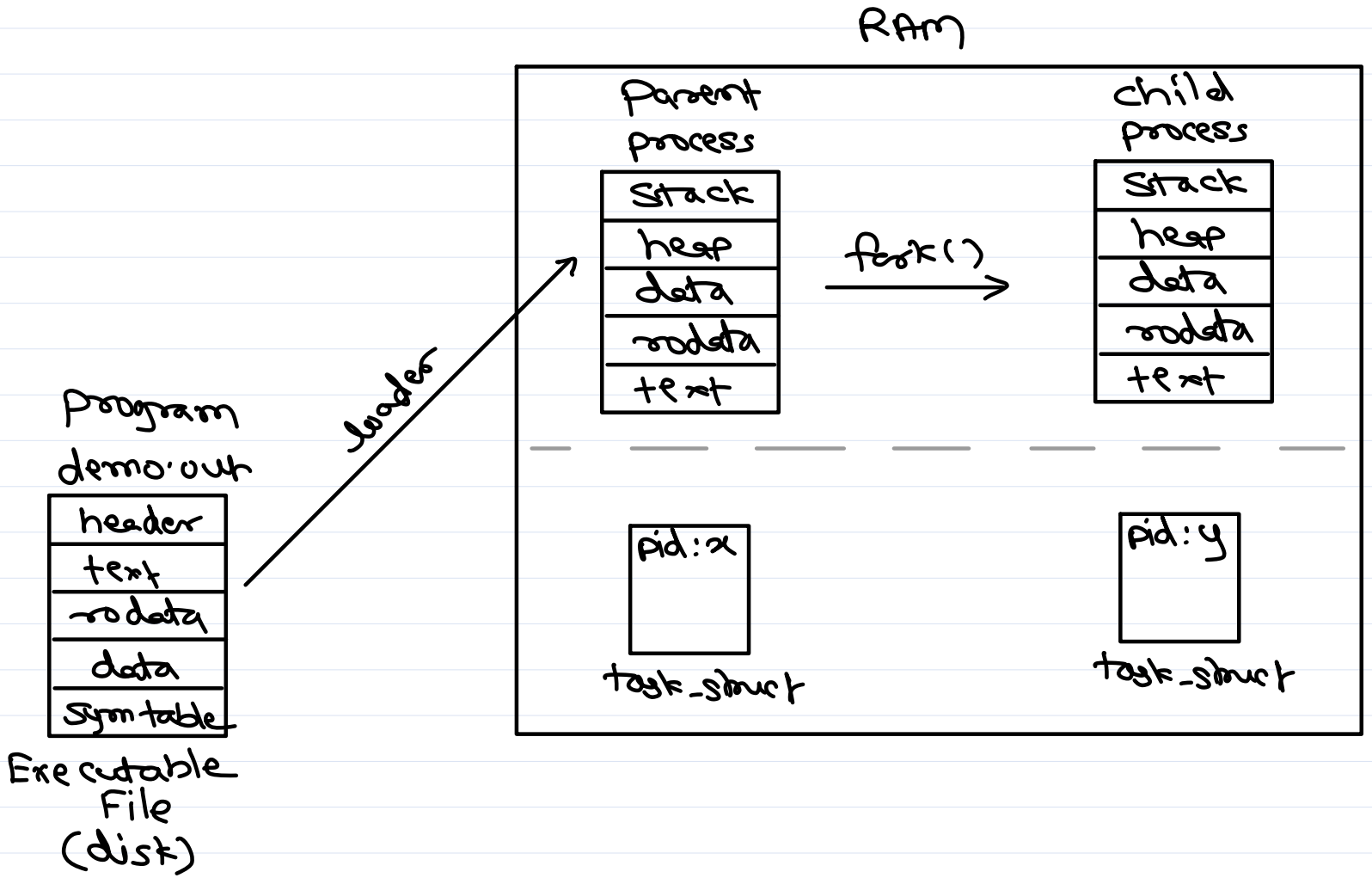
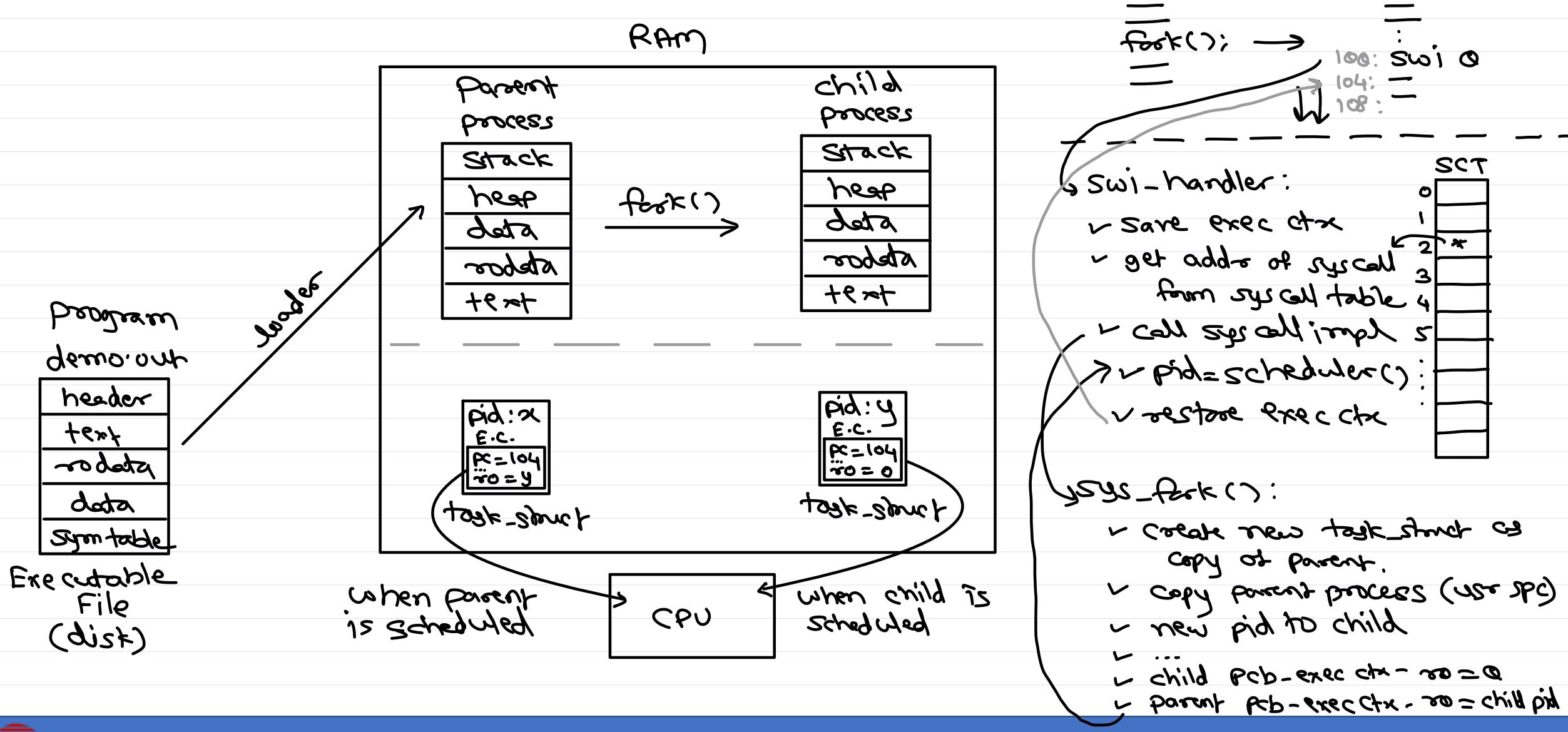
Embedded Operating Systems

Trainer: Nilesh Ghule

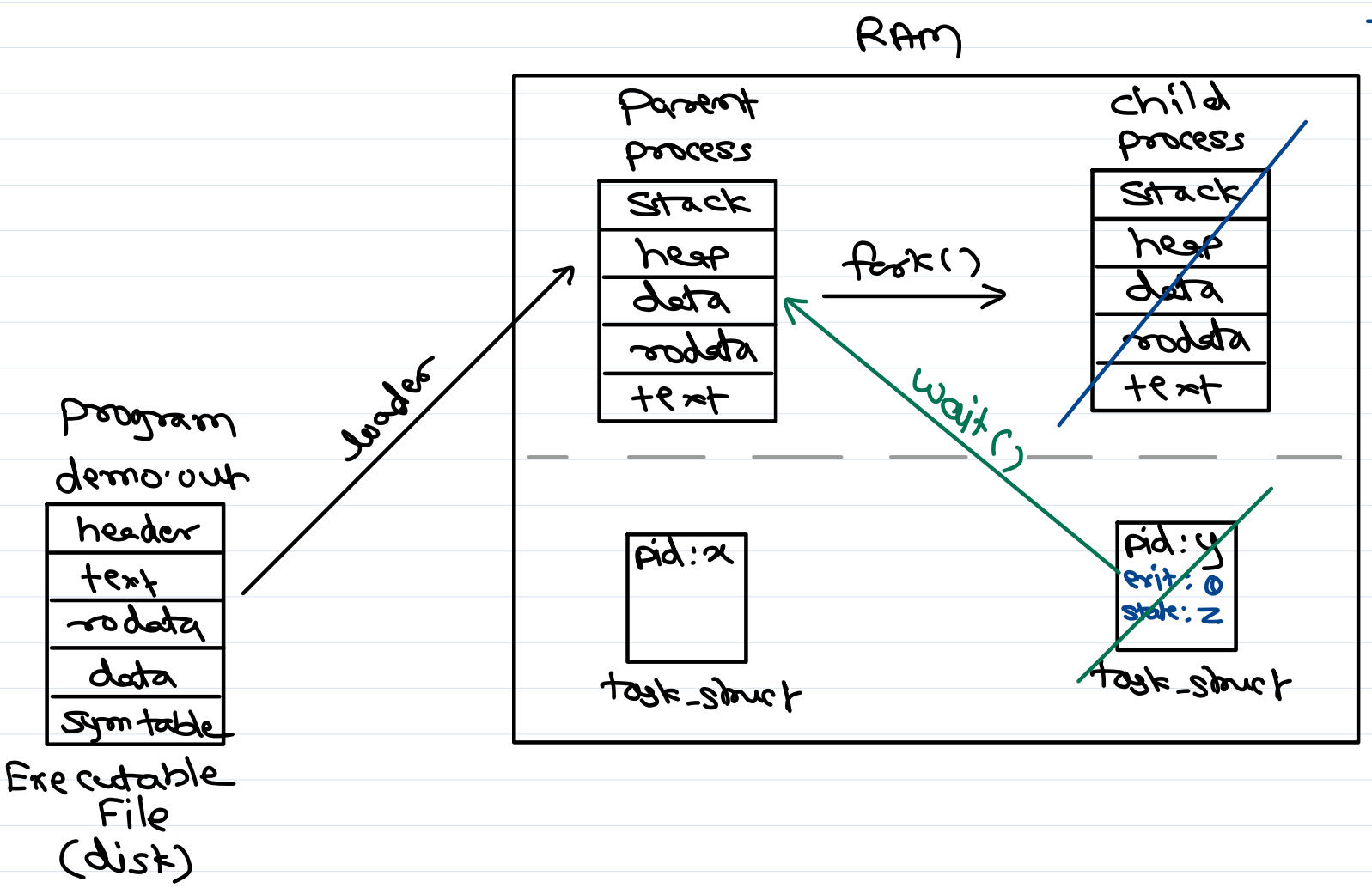
fork() system call



fork() system call



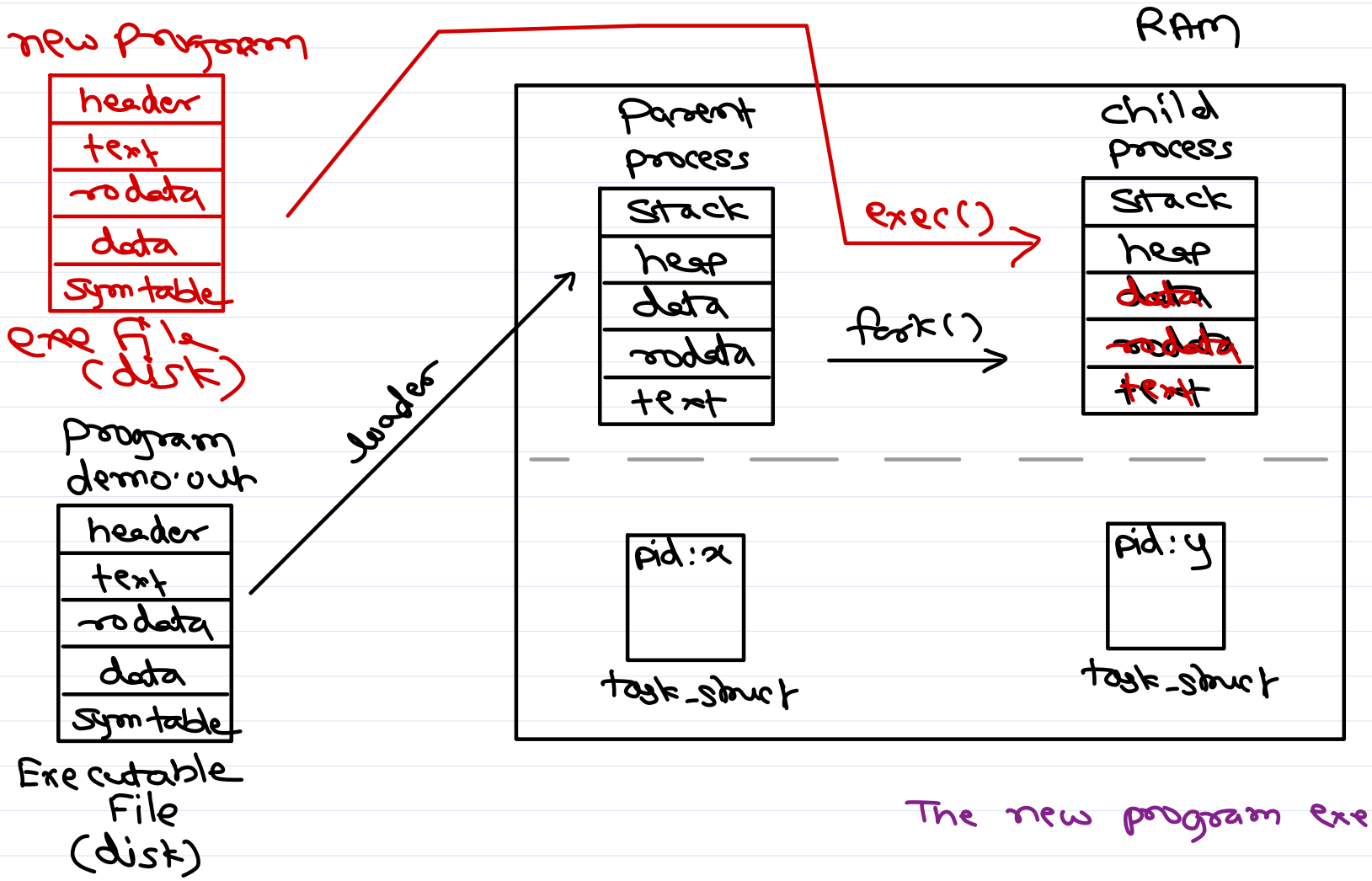
fork() system call



- exit(code):
  1. release resources occupied by current process e.g. user space memory, open files, ...
  2. write exit status in PCB of the current process.
  3. send SIGCHLD signal to the parent process.

wait(&status):
1. pause current process for termination of one of the child processes.
2. get exit status of child from its PCB into the out param.
3. release PCB of the child process

exec() system call




```
ret=fork();
if(ret==0){
    err=exec("/new/program",...);
    if(err<0){
        perror("exec() failed");
        exit(1);
    }
}
else{
    waitpid(ret, &s, 0);
    //...
}
```

exec() loads new program in calling process memory replacing old program image (section).

The new program execution begin from its entry point fn.

Thank you!

Nilesh Ghule <nilesh@sunbeaminfo.com>