xv6©-rev10

(Copyright Frans Kaashoek, Robert Morris, and Russ Cox.)

The exit syscall

Carmi Merimovich

Tel-Aviv Academic College

January 20, 2017

sys_exit

```
3766  sys_exit(void) {
        exit();
        return 0; // not reached
      }
```

exit

This is really a very simple system call

1. Release resources the process holds:
   - Close files.
   - We should have released user mode memory. xv6 DOES NOT do that.
2. Reparent child processes to process 1.
3. Declare event `myproc()->parent`.

The deallcation of the `proc` structure, kernel stack, and user memory is done at the parent.

exit (1)

```
2627  exit(void) {
      struct proc *p;
      int fd;

      if (myproc() == initproc)
        panic("init exiting");

      for (fd = 0; fd < NOFILE; fd++) {
        if (myproc->ofile[fd]) {
          fileclose(myproc()->ofile[fd]);
          myproc()->ofile[fd] = 0;
        }
      }

      begin_op(); iput(myproc()->cwd); end_op();
      myproc()->cwd = 0;
```

exit (2)

```
2649    acquire(&ptable.lock);
        wakeup1(myproc()->parent);
        for (p = ptable.proc;p < &ptable.proc[NPROC];p++) {
         if (p->parent == myproc()) {
          p->parent = initproc;
          if (p->state == ZOMBIE)
           wakeup1(initproc);
         }
        }

        myproc()->state = ZOMBIE;
        sched();
        panic("zombie exit");
       }
```