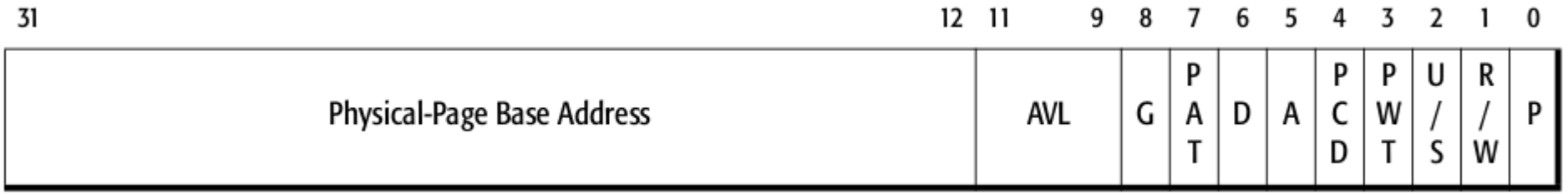
L5: Operating system structure

Nickolai Zeldovich
6.033 Spring 2012

Overall plan: enforce client/server modularity

- This lecture: enforcing modularity with an OS
- Next few lectures:
  - Allow client/server interaction between programs
  - Run multiple programs on a single CPU
  - Run multiple operating systems
  - Achieve good performance

x86 page table entry



- P: is this virtual page present?
- R/W: is this virtual page writable?
- U/S: is this virtual page accessible to user?

- No → MMU triggers a “page fault”

Unix abstractions

```
main() {
    chdir("/usr/rtm");

    int fd = open("quiz.txt", 0);

    char buf[512];
    int n = read(fd, buf, 512);
    write(1, buf, n);
    close(fd);
}
```

Kernel complexity

- 1975: Unix v6

  10,500 total lines of kernel code

- 2012: Linux 3.2

  300,000 lines: header files (data structures, APIs)
  490,000 lines: networking
  530,000 lines: sound
  700,000 lines: support for 60+ file systems
  1,880,000 lines: support for 25+ CPU architectures
  5,620,000 lines: drivers

  9,930,000 total lines of code

Summary

- Two key OS techniques:
  - Virtualization allows programs to share hardware
  - Abstractions provide portability, cooperation

- OS kernel enforces modularity:
  - Program vs program
  - Program vs kernel