 ***DEPARTMENT OF COMPUTER ENGINEERING***

1

Experiment No.

|  |  |
| --- | --- |
| Semester | S.E. Semester IV – Computer Engineering |
| Subject | Operating System |
| Subject Professor In-charge | SNA |
| Assisting Teachers | Ms. Rasika Ransing |
| Laboratory | M310B – Computer Engineering Laboratory |

|  |  |  |
| --- | --- | --- |
| Student Name | Akshay Prabhu | |
| Roll Number | 18102A0058 | |
| Grade and Subject Teacher’s Signature |  |  |

|  |  |  |
| --- | --- | --- |
| Experiment Number |  | |
| Experiment Title | Explore the following system calls: open, read, write, close | |
| Resources / Apparatus Required | Hardware: PC | Software:  Compiler |
| Objectives  (Skill Set / Knowledge Tested / Imparted) | System Calls | |
| Theory: | In computing, a system call is the programmatic way in which a computer program requests a service from the kernel of the operating system it is executed on. This may include hardware-related services (for example, accessing a hard disk drive), creation and execution of new processes, and communication with integral kernel services such as process scheduling. System calls provide an essential interface between a process and the operating system.  In most systems, system calls can only be made from userspace processes, while in some systems, OS/360 and successors for example, privileged system code also issues system calls  The architecture of most modern processors, with the exception of some embedded systems, involves a security model. For example, the rings model specifies multiple privilege levels under which software may be executed: a program is usually limited to its own address space so that it cannot access or modify other running programs or the operating system itself, and is usually prevented from directly manipulating hardware devices (e.g. the frame buffer or network devices).  However, many normal applications obviously need access to these components, so system calls are made available by the operating system to provide well-defined, safe implementations for such operations. The operating system executes at the highest level of privilege, and allows applications to request services via system calls, which are often initiated via interrupts. An interrupt automatically puts the CPU into some elevated privilege level, and then passes control to the kernel, which determines whether the calling program should be granted the requested service. If the service is granted, the kernel executes a specific set of instructions over which the calling program has no direct control, returns the privilege level to that of the calling program, and then returns control to the calling program. | |
| Code | //Constants  pub const \_SYSCTL: usize = 156;  pub const ACCEPT: usize = 43;  pub const ACCEPT4: usize = 288;  pub const ACCESS: usize = 21;  pub const ACCT: usize = 163;  pub const ADD\_KEY: usize = 248;  pub const ADJTIMEX: usize = 159;  pub const AFS\_SYSCALL: usize = 183;  pub const ALARM: usize = 37;  pub const ARCH\_PRCTL: usize = 158;  pub const BIND: usize = 49;  pub const BPF: usize = 321;  pub const BRK: usize = 12;  pub const CAPGET: usize = 125;  pub const CAPSET: usize = 126;  pub const CHDIR: usize = 80;  pub const CHMOD: usize = 90;  pub const CHOWN: usize = 92;  pub const CHROOT: usize = 161;  pub const CLOCK\_ADJTIME: usize = 305;  pub const CLOCK\_GETRES: usize = 229;  pub const CLOCK\_GETTIME: usize = 228;  pub const CLOCK\_NANOSLEEP: usize = 230;  pub const CLOCK\_SETTIME: usize = 227;  pub const CLONE: usize = 56;  pub const CLOSE: usize = 3;  pub const CONNECT: usize = 42;  pub const COPY\_FILE\_RANGE: usize = 326;  pub const CREAT: usize = 85;  pub const CREATE\_MODULE: usize = 174;  pub const DELETE\_MODULE: usize = 176;  pub const DUP: usize = 32;  pub const DUP2: usize = 33;  pub const DUP3: usize = 292;  pub const EPOLL\_CREATE: usize = 213;  pub const EPOLL\_CREATE1: usize = 291;  pub const EPOLL\_CTL: usize = 233;  pub const EPOLL\_CTL\_OLD: usize = 214;  pub const EPOLL\_PWAIT: usize = 281;  pub const EPOLL\_WAIT: usize = 232;  pub const EPOLL\_WAIT\_OLD: usize = 215;  pub const EVENTFD: usize = 284;  pub const EVENTFD2: usize = 290;  pub const EXECVE: usize = 59;  pub const EXECVEAT: usize = 322;  pub const EXIT: usize = 60;  pub const EXIT\_GROUP: usize = 231;  pub const FACCESSAT: usize = 269;  pub const FADVISE64: usize = 221;  pub const FALLOCATE: usize = 285;  pub const FANOTIFY\_INIT: usize = 300;  pub const FANOTIFY\_MARK: usize = 301;  pub const FCHDIR: usize = 81;  pub const FCHMOD: usize = 91;  pub const FCHMODAT: usize = 268;  pub const FCHOWN: usize = 93;  pub const FCHOWNAT: usize = 260;  pub const FCNTL: usize = 72;  pub const FDATASYNC: usize = 75;  pub const FGETXATTR: usize = 193;  pub const FINIT\_MODULE: usize = 313;  pub const FLISTXATTR: usize = 196;  pub const FLOCK: usize = 73;  pub const FORK: usize = 57;  pub const FREMOVEXATTR: usize = 199;  pub const FSETXATTR: usize = 190;  pub const FSTAT: usize = 5;  pub const FSTATFS: usize = 138;  pub const FSYNC: usize = 74;  pub const FTRUNCATE: usize = 77;  pub const FUTEX: usize = 202;  pub const FUTIMESAT: usize = 261;  pub const GET\_KERNEL\_SYMS: usize = 177;  pub const GET\_MEMPOLICY: usize = 239;  pub const GET\_ROBUST\_LIST: usize = 274;  pub const GET\_THREAD\_AREA: usize = 211;  pub const GETCPU: usize = 309;  pub const GETCWD: usize = 79;  pub const GETDENTS: usize = 78;  pub const GETDENTS64: usize = 217;  pub const GETEGID: usize = 108;  pub const GETEUID: usize = 107;  pub const GETGID: usize = 104;  pub const GETGROUPS: usize = 115;  pub const GETITIMER: usize = 36;  pub const GETPEERNAME: usize = 52;  pub const GETPGID: usize = 121;  pub const GETPGRP: usize = 18102A0058;  pub const GETPID: usize = 39;  pub const GETPMSG: usize = 181;  pub const GETPPID: usize = 110;  pub const GETPRIORITY: usize = 140;  pub const GETRANDOM: usize = 318;  pub const GETRESGID: usize = 120;  pub const GETRESUID: usize = 118;  pub const GETRLIMIT: usize = 97;  pub const GETRUSAGE: usize = 98;  pub const GETSID: usize = 124;  pub const GETSOCKNAME: usize = 51;  pub const GETSOCKOPT: usize = 55;  pub const GETTID: usize = 186;  pub const GETTIMEOFDAY: usize = 96;  pub const GETUID: usize = 102;  pub const GETXATTR: usize = 191;  pub const INIT\_MODULE: usize = 175;  pub const INOTIFY\_ADD\_WATCH: usize = 254;  pub const INOTIFY\_INIT: usize = 253;  pub const INOTIFY\_INIT1: usize = 294;  pub const INOTIFY\_RM\_WATCH: usize = 255;  pub const IO\_CANCEL: usize = 210;  pub const IO\_DESTROY: usize = 207;  pub const IO\_GETEVENTS: usize = 208;  pub const IO\_SETUP: usize = 206;  pub const IO\_SUBMIT: usize = 209;  pub const IOCTL: usize = 16;  pub const IOPERM: usize = 173;  pub const IOPL: usize = 172;  pub const IOPRIO\_GET: usize = 252;  pub const IOPRIO\_SET: usize = 251;  pub const KCMP: usize = 312;  pub const KEXEC\_FILE\_LOAD: usize = 320;  pub const KEXEC\_LOAD: usize = 246;  pub const KEYCTL: usize = 250;  pub const KILL: usize = 62;  pub const LCHOWN: usize = 94;  pub const LGETXATTR: usize = 192;  pub const LINK: usize = 86;  pub const LINKAT: usize = 265;  pub const LISTEN: usize = 50;  pub const LISTXATTR: usize = 194;  pub const LLISTXATTR: usize = 195;  pub const LOOKUP\_DCOOKIE: usize = 212;  pub const LREMOVEXATTR: usize = 198;  pub const LSEEK: usize = 8;  pub const LSETXATTR: usize = 189;  pub const LSTAT: usize = 6;  pub const MADVISE: usize = 28;  pub const MBIND: usize = 237;  pub const MEMBARRIER: usize = 324;  pub const MEMFD\_CREATE: usize = 319;  pub const MIGRATE\_PAGES: usize = 256;  pub const MINCORE: usize = 27;  pub const MKDIR: usize = 83;  pub const MKDIRAT: usize = 258;  pub const MKNOD: usize = 133;  pub const MKNODAT: usize = 259;  pub const MLOCK: usize = 149;  pub const MLOCK2: usize = 325;  pub const MLOCKALL: usize = 151;  pub const MMAP: usize = 9;  pub const MODIFY\_LDT: usize = 154;  pub const MOUNT: usize = 165;  pub const MOVE\_PAGES: usize = 279;  pub const MPROTECT: usize = 10;  pub const MQ\_GETSETATTR: usize = 245;  pub const MQ\_NOTIFY: usize = 244;  pub const MQ\_OPEN: usize = 240;  pub const MQ\_TIMEDRECEIVE: usize = 243;  pub const MQ\_TIMEDSEND: usize = 242;  pub const MQ\_UNLINK: usize = 241;  pub const MREMAP: usize = 25;  pub const MSGCTL: usize = 71;  pub const MSGGET: usize = 68;  pub const MSGRCV: usize = 70;  pub const MSGSND: usize = 69;  pub const MSYNC: usize = 26;  pub const MUNLOCK: usize = 150;  pub const MUNLOCKALL: usize = 152;  pub const MUNMAP: usize = 11;  pub const NAME\_TO\_HANDLE\_AT: usize = 303;  pub const NANOSLEEP: usize = 35;  pub const NEWFSTATAT: usize = 262;  pub const NFSSERVCTL: usize = 180;  pub const OPEN: usize = 2;  pub const OPEN\_BY\_HANDLE\_AT: usize = 304;  pub const OPENAT: usize = 257;  pub const PAUSE: usize = 34;  pub const PERF\_EVENT\_OPEN: usize = 298;  pub const PERSONALITY: usize = 135;  pub const PIPE: usize = 22;  pub const PIPE2: usize = 293;  pub const PIVOT\_ROOT: usize = 155;  pub const PKEY\_ALLOC: usize = 330;  pub const PKEY\_FREE: usize = 331;  pub const PKEY\_MPROTECT: usize = 329;  pub const POLL: usize = 7;  pub const PPOLL: usize = 271;  pub const PRCTL: usize = 157;  pub const PREAD64: usize = 17;  pub const PREADV: usize = 295;  pub const PREADV2: usize = 327;  pub const PRLIMIT64: usize = 302;  pub const PROCESS\_VM\_READV: usize = 310;  pub const PROCESS\_VM\_WRITEV: usize = 311;  pub const PSELECT6: usize = 270;  pub const PTRACE: usize = 101;  pub const PUTPMSG: usize = 182;  pub const PWRITE64: usize = 18;  pub const PWRITEV: usize = 296;  pub const PWRITEV2: usize = 328;  pub const QUERY\_MODULE: usize = 178;  pub const QUOTACTL: usize = 179;  pub const READ: usize = 0;  pub const READAHEAD: usize = 187;  pub const READLINK: usize = 89;  pub const READLINKAT: usize = 267;  pub const READV: usize = 19;  pub const REBOOT: usize = 169;  pub const RECVFROM: usize = 45;  pub const RECVMMSG: usize = 299;  pub const RECVMSG: usize = 47;  pub const REMAP\_FILE\_PAGES: usize = 216;  pub const REMOVEXATTR: usize = 197;  pub const RENAME: usize = 82;  pub const RENAMEAT: usize = 264;  pub const RENAMEAT2: usize = 316;  pub const REQUEST\_KEY: usize = 249;  pub const RESTART\_SYSCALL: usize = 219;  pub const RMDIR: usize = 84;  pub const RT\_SIGACTION: usize = 13;  pub const RT\_SIGPENDING: usize = 127;  pub const RT\_SIGPROCMASK: usize = 14;  pub const RT\_SIGQUEUEINFO: usize = 129;  pub const RT\_SIGRETURN: usize = 15;  pub const RT\_SIGSUSPEND: usize = 130;  pub const RT\_SIGTIMEDWAIT: usize = 128;  pub const RT\_TGSIGQUEUEINFO: usize = 297;  pub const SCHED\_GET\_PRIORITY\_MAX: usize = 146;  pub const SCHED\_GET\_PRIORITY\_MIN: usize = 147;  pub const SCHED\_GETAFFINITY: usize = 204;  pub const SCHED\_GETATTR: usize = 315;  pub const SCHED\_GETPARAM: usize = 143;  pub const SCHED\_GETSCHEDULER: usize = 145;  pub const SCHED\_RR\_GET\_INTERVAL: usize = 148;  pub const SCHED\_SETAFFINITY: usize = 203;  pub const SCHED\_SETATTR: usize = 314;  pub const SCHED\_SETPARAM: usize = 142;  pub const SCHED\_SETSCHEDULER: usize = 144;  pub const SCHED\_YIELD: usize = 24;  pub const SECCOMP: usize = 317;  pub const SECURITY: usize = 185;  pub const SELECT: usize = 23;  pub const SEMCTL: usize = 66;  pub const SEMGET: usize = 64;  pub const SEMOP: usize = 65;  pub const SEMTIMEDOP: usize = 220;  pub const SENDFILE: usize = 40;  pub const SENDMMSG: usize = 307;  pub const SENDMSG: usize = 46;  pub const SENDTO: usize = 44;  pub const SET\_MEMPOLICY: usize = 238;  pub const SET\_ROBUST\_LIST: usize = 273;  pub const SET\_THREAD\_AREA: usize = 205;  pub const SET\_TID\_ADDRESS: usize = 218;  pub const SETDOMAINNAME: usize = 171;  pub const SETFSGID: usize = 123;  pub const SETFSUID: usize = 122;  pub const SETGID: usize = 106;  pub const SETGROUPS: usize = 116;  pub const SETHOSTNAME: usize = 170;  pub const SETITIMER: usize = 38;  pub const SETNS: usize = 308;  pub const SETPGID: usize = 109;  pub const SETPRIORITY: usize = 141;  pub const SETREGID: usize = 114;  pub const SETRESGID: usize = 119;  pub const SETRESUID: usize = 117;  pub const SETREUID: usize = 113;  pub const SETRLIMIT: usize = 160;  pub const SETSID: usize = 112;  pub const SETSOCKOPT: usize = 54;  pub const SETTIMEOFDAY: usize = 164;  pub const SETUID: usize = 105;  pub const SETXATTR: usize = 188;  pub const SHMAT: usize = 30;  pub const SHMCTL: usize = 31;  pub const SHMDT: usize = 67;  pub const SHMGET: usize = 29;  pub const SHUTDOWN: usize = 48;  pub const SIGALTSTACK: usize = 131;  pub const SIGNALFD: usize = 282;  pub const SIGNALFD4: usize = 289;  pub const SOCKET: usize = 41;  pub const SOCKETPAIR: usize = 53;  pub const SPLICE: usize = 275;  pub const STAT: usize = 4;  pub const STATFS: usize = 137;  pub const STATX: usize = 332;  pub const SWAPOFF: usize = 168;  pub const SWAPON: usize = 167;  pub const SYMLINK: usize = 88;  pub const SYMLINKAT: usize = 266;  pub const SYNC: usize = 162;  pub const SYNC\_FILE\_RANGE: usize = 277;  pub const SYNCFS: usize = 306;  pub const SYSFS: usize = 139;  pub const SYSINFO: usize = 99;  pub const SYSLOG: usize = 103;  pub const TEE: usize = 276;  pub const TGKILL: usize = 234;  pub const TIME: usize = 201;  pub const TIMER\_CREATE: usize = 222;  pub const TIMER\_DELETE: usize = 226;  pub const TIMER\_GETOVERRUN: usize = 225;  pub const TIMER\_GETTIME: usize = 224;  pub const TIMER\_SETTIME: usize = 223;  pub const TIMERFD\_CREATE: usize = 283;  pub const TIMERFD\_GETTIME: usize = 287;  pub const TIMERFD\_SETTIME: usize = 286;  pub const TIMES: usize = 100;  pub const TKILL: usize = 200;  pub const TRUNCATE: usize = 76;  pub const TUXCALL: usize = 184;  pub const UMASK: usize = 95;  pub const UMOUNT2: usize = 166;  pub const UNAME: usize = 63;  pub const UNLINK: usize = 87;  pub const UNLINKAT: usize = 263;  pub const UNSHARE: usize = 272;  pub const USELIB: usize = 134;  pub const USERFAULTFD: usize = 323;  pub const USTAT: usize = 136;  pub const UTIME: usize = 132;  pub const UTIMENSAT: usize = 280;  pub const UTIMES: usize = 235;  pub const VFORK: usize = 58;  pub const VHANGUP: usize = 153;  pub const VMSPLICE: usize = 278;  pub const VSERVER: usize = 236;  pub const WAIT4: usize = 61;  pub const WAITID: usize = 247;  pub const WRITE: usize = 1;  pub const WRITEV: usize = 20;  //Syscalls  pub mod nr;  #[inline(always)]  pub unsafe fn syscall0(mut n: usize) -> usize {  asm!("syscall"  : "+{rax}"(n)  :  : "rcx", "r11", "memory"  : "volatile");  n  }  #[inline(always)]  pub unsafe fn syscall1(mut n: usize, a1: usize) -> usize {  asm!("syscall"  : "+{rax}"(n)  : "{rdi}"(a1)  : "rcx", "r11", "memory"  : "volatile");  n  }  #[inline(always)]  pub unsafe fn syscall2(mut n: usize, a1: usize, a2: usize) -> usize {  asm!("syscall"  : "+{rax}"(n)  : "{rdi}"(a1) "{rsi}"(a2)  : "rcx", "r11", "memory"  : "volatile");  n  }  #[inline(always)]  pub unsafe fn syscall3(mut n: usize, a1: usize, a2: usize, a3: usize) -> usize {  asm!("syscall"  : "+{rax}"(n)  : "{rdi}"(a1) "{rsi}"(a2) "{rdx}"(a3)  : "rcx", "r11", "memory"  : "volatile");  n  }  #[inline(always)]  pub unsafe fn syscall4(mut n: usize,  a1: usize,  a2: usize,  a3: usize,  a4: usize)  -> usize {  asm!("syscall"  : "+{rax}"(n)  : "{rdi}"(a1) "{rsi}"(a2) "{rdx}"(a3) "{r10}"(a4)  : "rcx", "r11", "memory"  : "volatile");  n  }  #[inline(always)]  pub unsafe fn syscall5(mut n: usize,  a1: usize,  a2: usize,  a3: usize,  a4: usize,  a5: usize)  -> usize {  asm!("syscall"  : "+{rax}"(n)  : "{rdi}"(a1) "{rsi}"(a2) "{rdx}"(a3) "{r10}"(a4) "{r8}"(a5)  : "rcx", "r11", "memory"  : "volatile");  n  }  #[inline(always)]  pub unsafe fn syscall6(mut n: usize,  a1: usize,  a2: usize,  a3: usize,  a4: usize,  a5: usize,  a6: usize)  -> usize {  asm!("syscall"  : "+{rax}"(n)  : "{rdi}"(a1) "{rsi}"(a2) "{rdx}"(a3) "{r10}"(a4) "{r8}"(a5)"{r9}"(a6)  : "rcx", "r11", "memory"  : "volatile");  n  } | |