Resa Fajar Sukma – 1313618029

Operating System

#Catatan jika menemukan line yang diblok dengan warna hijau, itu adalah line yang ditambahkan/mengubah kode lama.

==========

Makefile

Perubahan

- CS333_PROJECT ?= (0 / 1)
- PRINT_SYSCALLS ?= (0/1)

System Call Tracing

Perubahan pada file “syscall.c” → fungsi yang ditambahkan baris kodenya “syscall()”

```
Void
syscall(void)
{
  int num;
  struct proc *curproc = myproc();

  num = curproc->tf->eax;
  if(num > 0 && num < NELEM(syscalls) && syscalls[num]) {
    curproc->tf->eax = syscalls[num]();
    #ifdef PRINT_SYSCALLS
    cprintf("%s -> %d\n",
            syscallnames[num], curproc->tf->eax);
    #endif // PRINT_SYSCALLS
  } else {
    cprintf("%d %s: unknown sys call %d\n",
            curproc->pid, curproc->name, num);
    curproc->tf->eax = -1;
  }
}
```

Date System Call

Yang diubah

- user.h
- usys.S

- syscall.c
- sysproc.c

user.h, menambahkan baris kode dibawah

```
#ifdef CS333_P1
int date(struct rtcdate *);
#endif // CS333_P1
```

usys.S, menambahkan baris kode dibawah

```
SYSCALL(date)
```

syscall.c, menambahkan baris kode dibawah

Menambahkan ini pada array syscallnames

```
[SYS_date]    "date"
```

Menambahkan ini pada array syscalls

```
#ifdef CS333_P1
[SYS_date]    sys_date,
#endif // CS333_P1
```

Menambahkan pada bagian “extern”

```
#ifdef CS333_P1
extern int sys_date(void);
#endif // CS333_P1
```

sysproc.c, menambahkan baris kode dibawah

```
#ifdef CS333_P1
int
sys_date(void)
{
  struct rtcdate *d;
  if(argptr(0, (void*)&d, sizeof(struct rtcdate)) < 0)
    return -1;
  cmostime(d);
  return 0;
}
#endif // CS333_P1
```

Mengubah Makefile pada bagian

```
ifeq ($(CS333_PROJECT), 1)
CS333_CFLAGS += -DCS333_P1
CS333_UPROGS += _date
endif

awalnya adalah
CS333_UPROGS += #_date
```

Control-P (Process Information)

Perubahan ada pada file “proc.c” dan “proc.h”

proc.h

```
// Per-process state
struct proc {
  uint sz;                     // Size of process memory (bytes)
  pde_t* pgdir;                // Page table
  char *kstack;                // Bottom of kernel stack for this process
  enum procstate state;        // Process state
  uint pid;                    // Process ID
  struct proc *parent;         // Parent process. NULL indicates no parent
  struct trapframe *tf;        // Trap frame for current syscall
  struct context *context;     // swtch() here to run process
  void *chan;                  // If non-zero, sleeping on chan
  int killed;                  // If non-zero, have been killed
  struct file *ofile[NOFILE];  // Open files
  struct inode *cwd;           // Current directory
  char name[16];               // Process name (debugging)
  uint start_ticks;            // Untuk menyimpan waktu pembuatan proc
};
```

proc.c

- **Pada function “allocproc()”, menambahkan baris kode**

  ```
  p->start_ticks = ticks;
  ```

- **Pada function “procdumpP1()”, mengganti baris kode lama menjadi baris kode baru**

  ```
  void
  procdumpP1(struct proc *p, char *state_string)
  {
    uint now = (ticks - p->start_ticks);
  ```

```
  uint a = now/1000;
  uint b = now%1000;
  cprintf("%d\t%s\t\t%d.%d\t%s\t%d", p->pid, p->name, a, b, state_string, p->sz);
  return;
}
```

#Sebelum Diubah

```
void
procdumpP1(struct proc *p, char *state_string)
{
  cprintf("TODO for Project 1, delete this line and implement procdumpP2P3P4() in
proc.c to print a row\n");
  return;
}
```