Code Modification Report

Nama : Muhammad Anindyo Poetra Mufatyta

NIM : 1313619004

Prodi : Ilmu Komputer 2019

User.h

```
31 #ifdef CS333_P2
32 uint getuid(void); // UID of the parent process
33 uint getgid(void); // GID of the parent process
34 uint getppid(void); // process ID of the parent process
35 int setuid(uint); // set UID
36 int setgid(uint); // set GID
37 int getprocs(uint max, struct uproc* table);
38 #endif // CS333_P2
```

Usys.S

Line 34-39

```
34 SYSCALL(getuid) #project2
35 SYSCALL(getgid)
36 SYSCALL(getppid)
37 SYSCALL(setuid)
38 SYSCALL(setgid)
39 SYSCALL(getprocs)
```

Syscall.c

Line 112-119

```
112 #ifdef CS333_P2
113 extern int sys_getuid(void);
114 extern int sys_getgid(void);
115 extern int sys_getppid(void);
116 extern int sys_setuid(void);
117 extern int sys_setgid(void);
118 extern int sys_getprocs(void);
119 #endif // CS333_P2
```

Line 149-156

```
149 #ifdef CS333_P2
150 [SYS_getuid]  sys_getuid,
151 [SYS_getgid]  sys_getgid,
152 [SYS_getppid] sys_getppid,
153 [SYS_setuid]  sys_setuid,
154 [SYS_setgid]  sys_setgid,
155 [SYS_getprocs]  sys_getprocs,
156 #endif // CS333_P2
```

Line 188-195

```
188 #ifdef CS333_P2
189   [SYS_getuid]  "getuid",
190   [SYS_getgid]  "getgid",
191   [SYS_getppid] "getppid",
192   [SYS_setuid]  "setuid",
193   [SYS_setgid]  "setgid",
194   [SYS_getprocs]  "getprocs",
195 #endif // CS333_P2
```

Sysproc.c

Line 101-176

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

//project 2
#ifdef CS333_P2
int
sys_getuid(void)
{
  return myproc()->uid;
}

int
sys_getgid(void)
{
  return myproc()->gid;
}
```

```
int
sys_getppid(void)
{
  if(myproc()->parent == NULL)
    return myproc()->pid;
  else
    return myproc()->parent->pid;
}

int
sys_setuid(void)
{
  int test;
  if(argint(0, &test)<0)
    return -1;
  if(test < 0 || test >32767)
    return -1;
  else{
    myproc()->uid = test;
    return 0;
  }
}

int
sys_setgid(void)
{
  int test;
  if(argint(0, &test)<0)
    return -1;
  if(test < 0 || test >32767)
    return -1;
  else{
    myproc()->gid = test;
    return 0;
  }
}

int
sys_getprocs(void)
{
  struct uproc *p;
  int max;

  if(argint(0,&max)<0){
    return -1;
  }
  if(argptr(1, (void*)&p, sizeof(struct uproc) * max) < 0)
    return -1;
```

```
   return getprocs(max, p);
 }
 #endif // CS333_P2
```

Proc.h

Line 54-59

```
54 #ifdef CS333_P2 //project2
55   uint uid;
56   uint gid;
57   uint cpu_ticks_total;
58   uint cpu_ticks_in;
59 #endif // CS333_P2
```

Proc.c

Line 9-11

```
9 #ifdef CS333_P2
10 #include "uproc.h"
11 #endif // CS33_P2
```

Line 157-160

```
157 #ifdef CS333_P2 //project2
158   p->cpu_ticks_total = 0;
159   p->cpu_ticks_in = 0;
160 #endif // CS333_P2
```

Line 188-191

```
188   #ifdef CS333_P2
189   p->uid = DEFAULT_UID;
190   p->gid = DEFAULT_GID;
191   #endif // CS333_P2
```

Line 252-255

```
252   #ifdef CS333_P2
253   np->uid = curproc->uid;
254   np->gid = curproc->gid;
255   #endif //CS333_P2
```

Line 410-412

```
410 #ifdef CS333_P2
```

```
411         p->cpu_ticks_in = ticks;
412         #endif // CS333_P2
```

Line 453-455

```
453     #ifdef CS333_P2
454     p->cpu_ticks_total += (ticks - p->cpu_ticks_in);
455     #endif // CS333_P2
```

Line 583-627

```
 uint elapsed_s;
   uint elapsed_ms;

   elapsed_ms = ticks - p->start_ticks;
   elapsed_s = elapsed_ms / 1000;
   elapsed_ms = elapsed_ms % 1000;

   uint elapsed_cpu_s;
   uint elapsed_cpu_ms;
   uint ppid;
   if(p->parent){
     ppid = p->parent->pid;
   }
   else{
     ppid = p->pid;
   }

   elapsed_cpu_ms = p->cpu_ticks_total;
   elapsed_cpu_s = elapsed_cpu_ms / 1000;
   elapsed_cpu_ms = elapsed_cpu_ms % 1000;

   char* zero = "";
   if(elapsed_ms < 100 && elapsed_ms >= 10)
     zero = "0";
   if(elapsed_ms < 10)
     zero = "00";

   char* cpu_zero = "";
   if(elapsed_cpu_ms < 100 && elapsed_cpu_ms >= 10)
     cpu_zero = "0";
   if(elapsed_cpu_ms < 10)
     cpu_zero = "00";

   cprintf(
     "\n%d\t%s\t%s%d\t%s%d\t%s%d\t%d.%s%d\t%d.%s%d\t%s\t%d\t",
     p->pid,
     p->name, "      ",
     p->uid, "        ",
```

```
    p->gid, "",
    ppid,
    elapsed_s, zero, elapsed_ms,
    elapsed_cpu_s, cpu_zero, elapsed_cpu_ms,
    state_string,
    p->sz
  );
```

Line 1000-1033

```
#ifdef CS333_P2
int
getprocs(uint max, struct uproc* upTable){
  struct proc* p;
  int procsNumber = 0;
  acquire(&ptable.lock);

  for(p = ptable.proc; p < &ptable.proc[NPROC]; p++){
    if (procsNumber < max) {
      if(p->state != UNUSED && p->state != EMBRYO) {
        if(p->state >= 0 && p->state < NELEM(states) && states[p->state]){
          safestrcpy(upTable[procsNumber].state, states[p->state],STRMAX);
        } else {
          safestrcpy(upTable[procsNumber].state,"???",STRMAX);
        }

        upTable[procsNumber].pid = p->pid;
        upTable[procsNumber].uid = p->uid;
        upTable[procsNumber].gid = p->gid;
        upTable[procsNumber].ppid = p->parent ? p->parent->pid : p->pid;
        upTable[procsNumber].elapsed_ticks = ticks - p->start_ticks;
        upTable[procsNumber].CPU_total_ticks = p->cpu_ticks_total;
        upTable[procsNumber].size = p->sz;
        safestrcpy(upTable[procsNumber].name, p->name, STRMAX);
        procsNumber++;
      }
    } else {
      break;
    }
  }
  release(&ptable.lock);
  return procsNumber;
}
#endif // CS333_P2
```

Defs.h

Line 1-3

```
1 #ifdef CS333_P2
2 #include "uproc.h"
3 #endif
```

Line 130-132

```
130 #ifdef CS333_P2
131 int             getprocs(uint max, struct uproc* upTable);
132 #endif // CS333_P2
```

Ps.c

File program baru

```
1 #ifdef CS333_P2
 2 #include "types.h"
 3 #include "user.h"
 4 #include "uproc.h"
 5
 6 #define MAX 16
 7
 8 int
 9 main(void)
10 {
11   struct uproc *proc = malloc(sizeof(struct uproc)*MAX);
12   int proc_num = getprocs(MAX, proc);
13   printf(1,"PID\tName\t\tUID\tGID\tPPID\tElapsed\tCPU\tState\tSize\n");
14
15   int i;
16   for(i = 0; i<proc_num; i++){
17     struct uproc current_proc = proc[i];
18     uint elapsed_ticks = current_proc.elapsed_ticks;
19     uint elapsed_s = elapsed_ticks/1000;
20     uint elapsed_ms = elapsed_ticks%1000;
21
22     uint elapsed_cpu_ticks = current_proc.CPU_total_ticks;
23     uint elapsed_cpu_s = elapsed_cpu_ticks/1000;
24     uint elapsed_cpu_ms = elapsed_cpu_ticks % 1000;
25
26     char* zero = "";
27     if(elapsed_ms < 100 && elapsed_ms >= 10)
28       zero = "0";
29     if(elapsed_ms < 10)
30       zero = "00";
31
32     char* cpu_zero = "";
33     if(elapsed_cpu_ms < 100 && elapsed_cpu_ms >= 10)
34       cpu_zero = "0";
```

```
35      if(elapsed_cpu_ms < 10)
36        cpu_zero = "00";
37
38      printf(
39        1,
40        "%d\t%s\t\t%d\t%d\t%d\t%d.%s%d\t%d.%s%d\t%s\t%d\n",
41        current_proc.pid,
42        current_proc.name,
43        current_proc.uid,
44        current_proc.gid,
45        current_proc.ppid,
46        elapsed_s, zero, elapsed_ms,
47        elapsed_cpu_s, cpu_zero, elapsed_cpu_ms,
48        current_proc.state,
49        current_proc.size
50      );
51    }
52
53    free(proc);
54    exit();
55  }
56  #endif
```

Time.c

File program baru

```
1  #ifdef CS333_P2
2  #include "types.h"
3  #include "user.h"
4
5  int main(int argc, char *argv[]){
6       if(argc == 1) {
7         printf(1, "(null) ran in 0.00\n");
8       } else {
9         int start = uptime();
10        int pid = fork();
11
12        if (pid > 0) {
13          pid = wait();
14        } else if (pid == 0) {
15          exec(argv[1], argv+1);
16          printf(1, "ERROR: Unknown Command\n");
17          kill(getppid());
18          exit();
19        } else {
20          printf(1, "ERROR: Fork error return -1\n");
21        }
22
```

```
23        int end = uptime();
24        int timelapse = end - start;
25        int seconds = timelapse/1000;
26        int ms = timelapse%1000;
27        char *msZeros = "";
28
29        if (ms < 10) {
30          msZeros = "00";
31        } else if (ms < 100) {
32          msZeros = "0";
33        }
34
35        printf(
36          1,
37          "%s ran in %d.%s%d\n",
38          argv[1],
39          seconds,
40          msZeros,
41          ms
42        );
43      }
44      exit();
45 }
46 #endif // CS333_P2
```